1-1-2006

# A Unified Alert Fusion Model For Intelligent Analysis Of Sensor Data In An Intrusion Detection Environment

Ambareen Siraj

# A UNIFIED ALERT FUSION MODEL FOR INTELLIGENT ANALYSIS OF SENSOR DATA IN AN INTRUSION DETECTION ENVIRONMENT

By

Ambareen Siraj

A Dissertation
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Computer Science
in the Department of Computer Science and Engineering

Mississippi State, Mississippi

August 2006

A UNIFIED ALERT FUSION MODEL FOR INTELLIGENT ANALYSIS OF

SENSOR DATA IN AN INTRUSION DETECTION ENVIRONMENT

By

Ambareen Siraj

Approved:

| | |
|---|---|
| Dr. Rayford B. Vaughn<br>Professor of Computer Science & Engineering<br>(Major Professor) | Dr. Susan M. Bridges<br>Professor of Computer Science & Engineering<br>(Committee Member) |
| Dr. Julia E. Hodges<br>Professor of Computer Science & Engineering<br>(Committee Member) | Dr. David A. Dampier<br>Assistant Professor of Computer Science & Engineering<br>(Committee Member) |
| Dr. Jeffrey Carver<br>Assistant Professor of Computer Science & Engineering<br>(Committee Member) | Dr. Kirk Schulz<br>Dean of Bagley College of Engineering |

Dr. Edward B. Allen
Associate Professor and
Graduate Coordinator of
Computer Science & Engineering

Name: Ambareen Siraj

Date of Degree: August 5, 2006

Institution: Mississippi State University

Major Field: Computer Science

Major Professor: Dr. Rayford B. Vaughn

Title of Study: A UNIFIED ALERT FUSION MODEL FOR INTELLIGENT ANALYSIS OF SENSOR DATA IN AN INTRUSION DETECTION ENVIRONMENT

Pages in Study: 234

Candidate for Doctor of Philosophy

The need for higher-level reasoning capabilities beyond low-level sensor abilities has prompted researchers to use different types of sensor fusion techniques for better situational awareness in the intrusion detection environment. These techniques primarily vary in terms of their mission objectives. Some prioritize alerts for alert reduction, some cluster alerts to identify common attack patterns, and some correlate alerts to identify multi-staged attacks. Each of these tasks has its own merits. Unlike previous efforts in this area, this dissertation combines the primary tasks of sensor alert fusion, i.e., alert prioritization, alert clustering and alert correlation into a single framework such that individual results are used to quantify a confidence score as an overall assessment for global diagnosis of a system's security health. Such a framework is especially useful in a multi-sensor environment where the sensors can collaborate with or complement each other to provide increased reliability, making it essential that the outputs of the sensors

are fused in an effective manner in order to provide an improved understanding of the security status of the protected resources in the distributed environment.

This dissertation uses a possibilistic approach in intelligent fusion of sensor alerts with Fuzzy Cognitive Modeling in order to accommodate the impreciseness and vagueness in knowledge-based reasoning. We show that our unified architecture for sensor fusion provides better insight into the security health of systems. A new multi-level alert clustering method is developed to accommodate inexact matching in alert features and is shown to provide relevance to more alerts than traditional exact clustering. Alert correlation with a new abstract incident modeling technique is shown to deal with scalability and uncertainty issues present in traditional alert correlation. New concepts of dynamic fusion are presented for overall situation assessment, which a) in case of misuse sensors, combines results of alert clustering and alert correlation, and b) in case of anomaly sensors, corroborates evidence from primary and secondary sensors for deriving the final conclusion on the systems' security health.

DEDICATION

To my father, Muhammad Sirajuddin, whose unfulfilled dream guided me this way and to my mother, Alo Siraj, whose unconditional love sustained me this way.

ACKNOWLEDGEMENTS

I would like to express my sincerest appreciation for the people who helped me in many ways during the course of this dissertation.

My earnest gratitude goes to my major professor and dissertation director, Dr. Rayford B. Vaughn, who guided me throughout my academic term here at Mississippi State University (MSU) with constant encouragement and imperative directions and had the patience to accommodate my mistakes to make me learn from them. I am grateful for the generous financial support and the teaching and research opportunities he has provided throughout my appointment as research assistant in the Center for Computer Security Research (CCSR) at MSU.

I am extremely grateful to Dr. Susan M. Bridges, member of my graduate committee, for her invaluable advice and gracious help throughout my graduate studies and the course of this dissertation. I would also like to gratefully acknowledge Dr. David A. Dampier, Dr. Julia E. Hodges, Dr. Thomas Philip and Dr. Jeffrey Carver for serving as members in my graduate committee, and providing me with helpful suggestions and comments. I am also thankful to Dr. Julia E. Hodges for providing me with financial support during my appointment as teaching assistant in the Department of Computer Science and Engineering at MSU.

I would like to take this opportunity to acknowledge the support for infrastructure and resources needed in this work as provided by the Department of Computer Science and Engineering at MSU. A special appreciation goes to fellow CCSR student, Wesley McGrew for his effort with a difficult experiment. In addition, I would like to acknowledge the interest and assistance of other former/current students in the CCSR, specially, Shvetank Jain for his help with sensor report generation, and Wei Lei, German Florez, and Zhen Liu with whom I have worked closely during the span of this research. I would like to specially acknowledge the friendly staff at the Department of Computer Science and Engineering (Keri Chisolm, Russ Ward, Casey West, Brenda Collins, Jo Coleson, and Brandi Velcek), who was always willing to help me with anything and everything that I needed during my time at MSU. I wish to thank all the faculty members at the Department of Computer Science and Engineering whom I have come across and learned my way into the world of teaching and research.

On a personal note, I wish to express my heartfelt gratitude to my spouse, Sheikh, and my son, Reetesh, without whose love, sacrifice & support, I would not come this far.

Finally, I gratefully acknowledge the following agencies and organizations that have partially funded my research at MSU during different times over the last few years:

- The Army Research Laboratory (Contract # DAAD 17-01-C-001101010015);
- The Office of Naval Research and Missile Defense Agency (Grant # N00014-01-1-0678);
- National Science Foundation (Grant # CCR-0098024, CCR-0085749, CCR-998852, and SCI-0430354); and
- The National Security Agency/DoD (Grant # H98230-04-1-0205).

TABLE OF CONTENTS


Page

DEDICATION ..... ii

ACKNOWLEDGEMENTS ..... iii

LIST OF TABLES ..... viii

LIST OF FIGURES ..... xi

CHAPTER

I. INTRODUCTION ..... 1

1.1 Information Assurance, Intrusion Detection and Sensor Alert Fusion. 1
1.2 Sensor Fusion Tasks ..... 3
1.3 Multi-Sensor Environment ..... 4
1.4 Motivation ..... 6
1.5 Design Goals ..... 9
1.6 Hypothesis ..... 11
1.7 Contribution ..... 12
1.8 Terms ..... 13
1.9 Organization ..... 15

II. LITERATURE REVIEW ..... 16

2.1 Sensor Fusion Approaches ..... 16
2.2 Selected Research based on Reasoning under Certainty for Alert Fusion ..... 18
2.2.1 Aggregation and Correlation of Alerts ..... 18
2.2.2 Alert Clustering with Abstraction ..... 20
2.2.3 Co-operative Intrusion Detection ..... 22
2.2.4 Prerequisite - Consequence Model of Alert Correlation ..... 24
2.2.5 Alert Fusion Framework for Scenario Recognition ..... 27
2.2.6 Hybrid Intrusion Data Fusion ..... 29

CHAPTER Page

2.3 Selected Research based on Reasoning under Uncertainty for Alert Fusion ..... 30
2.3.1 Probabilistic Approach ..... 30
2.3.1.1 Alert Correlation based on Probabilistic Theory ..... 30
2.3.1.2 Building Scenarios from Heterogeneous Alert Stream .. 33
2.3.1.3 Alert Correlation with Hidden Colored Petri-Nets ..... 35
2.3.1.4 Alert Fusion based on Intrusion Reference Modeling ... 37
2.3.1.5 Probabilistic Intrusion Data Fusion ..... 39
2.3.2 Possibilistic Approach ..... 41
2.3.2.1 Our Previous Work using Fuzzy Cognitive Modeling for Decision Support in Network Security ..... 41
2.3.2.2 Fuzzy Intrusion Recognition Engine ..... 45

III. RESEARCH APPROACH ..... 47

3.1 Overview ..... 47
3.2 The Unified Alert Fusion Model ..... 47
3.2.1 Alert Prioritization ..... 49
3.2.2 Association Assessment ..... 51
3.2.2.1 Alert Clustering ..... 52
3.2.2.2 Alert Correlation ..... 66
3.2.3 Situation Assessment ..... 78
3.3 Summary ..... 90

IV. EXPERIMENTS AND RESULTS ..... 91

4.1 Experimental Design ..... 91
4.2 Experimental Setup ..... 93
4.3 Experimental Results ..... 100
4.3.1 Alert Prioritization Experiment ..... 101
Objective ..... 101
Evaluation ..... 101
Results and Analysis (R&A) ..... 102
Summary of Alert Prioritization Experiment ..... 105
4.3.2 Alert Correlation Experiment ..... 105
Objective ..... 105
Evaluation ..... 106
Results and Analysis (R&A) ..... 107
R&A for RealSecure-NCSU Sensor Report ..... 107
R&A for RealSecure-MSU Sensor Report ..... 117
R&A for Snort-MSU Sensor Report ..... 126
R&A for MultiSensor-MSU Report ..... 134
Summary for Alert Correlation Experiment ..... 142

CHAPTER Page

4.3.3 Alert Clustering Experiment ..... 143
Objective ..... 143
Evaluation ..... 143
Results and Analysis (R&A) ..... 145
R&A for RealSecure-NCSU Sensor Report ..... 146
R&A for RealSecure-MSU Sensor Report ..... 154
R&A for Snort-MSU Sensor Report ..... 161
R&A for MultiSensor-MSU Report ..... 171
Summary for Alert Clustering Experiment ..... 182
4.3.4 Misuse Situation Assessment Experiment ..... 183
Objective ..... 183
Results and Analysis (R&A) ..... 183
R&A for RealSecure-NCSU Sensor Report ..... 184
R&A for RealSecure-MSU Sensor Report ..... 189
R&A for Snort-MSU Sensor Report ..... 193
R&A for MultiSensor-MSU Report ..... 197
Summary for Misuse Situation Assessment Experiment ..... 202
4.3.5 Anomaly Situation Assessment Experiment ..... 203
Objective ..... 203
Results and Analysis (R&A) ..... 203
Summary for Anomaly Situation Assessment Experiment ..... 207
4.4 Summary of Results ..... 208

IV. CONCLUSIONS AND FUTURE WORK ..... 211

5.1 Contributions and Summary ..... 211
5.2 Limitations and Future Works ..... 216
5.3 Related Publications ..... 220

BIBLIOGRAPHY ..... 222

APPENDICES

A. Comparative Summary of the Literature Review ..... 227

B. Categorization of Attacks for Realsecure and Snort Sensors ..... 231

C. Criticality Indexes of Source/Target Communication and Attacks ..... 236

LIST OF TABLES


| TABLE | | Page |
|---|---|---|
| 1.1 | Terms Related to the Research | 14 |
| 3.1 | Features of Suspicious Clusters | 61 |
| 4.1 | Alert Correlation Performance for the RealSecure-NCSU Sensor Report | 109 |
| 4.2 | MSU and NCSU Alert Correlation Results for the RealSecure-NCSU Sensor Report | 113 |
| 4.3 | Incident Association Assessment for the RealSecure-NCSU Sensor Report | 115 |
| 4.4 | Correlation Performance for the RealSecure-MSU Sensor Report | 118 |
| 4.5 | Incident Situation for Host mill from analyzing the RealSecure-NCSU and the RealSecure-MSU Sensor Reports | 121 |
| 4.6 | Incident Association Assessment for RealSecure-MSU Sensor Report | 124 |
| 4.7 | Correlation Performance for the Snort-MSU Sensor Report | 126 |
| 4.8 | Incident Association Assessment for the Snort-MSU Sensor Report | 130 |
| 4.9 | Correlation Performance for the MultiSensor-MSU Report | 134 |
| 4.10 | Comparison of Correlated Alerts found for RealSecure-MSU Report, Snort-MSU Report and MultiSensor-MSU Report | 135 |
| 4.11 | Comparison of Incident Situation Discovered after Analyzing RealSecure-MSU Report, Snort-MSU Report, and MultiSensor-MSU Report | 137 |
| 4.12 | Comparison of Incident Situation for Host mill analyzing LLDOS 2.0.2 Inside Zone Dataset of RealSecure-MSU Report, Snort-MSU Report, and MultiSensor-MSU Report | 140 |

TABLE Page

4.13 Cluster Association Assessment for RealSecure-NCSU Sensor Report ..... 147

4.14 Cluster Association of Host hume for the RealSecure-NCSU Sensor Report..... 151

4.15 Cluster Association Assessment for the RealSecure-MSU Sensor Report ... 156

4.16 Cluster Association for LLDOS 1.0 Inside Zone Dataset analyzing the RealSecure-MSU Sensor Report..... 157

4.17 Cluster Association of Host mill for RealSecure-MSU Sensor Report..... 158

4.18 Cluster Association Assessment for Snort-MSU Sensor Report..... 163

4.19 Cluster Association of Host falcon for Snort-MSU Sensor Report..... 165

4.20 Cluster Association of Host locke for the Snort-MSU Sensor Report ..... 167

4.21 Cluster Association of Host mill for the Snort-MSU Sensor Report ..... 168

4.22 Cluster Association of Host swallow for the Snort-MSU Sensor Report..... 170

4.23 Cluster Association Assessment for MultiSensor-MSU Report..... 173

4.24 Cluster Association of Host pascal for the MultiSensor-MSU Report..... 175

4.25 Cluster Association of Host marx for the MultiSensor-MSU Report ..... 177

4.26 Cluster Association of Host swan for the MultiSensor-MSU Report ..... 178

4.27 Cluster Association of Host mill for the MultiSensor-MSU Report ..... 180

4.28 Situation Assessment for the RealSecure-NCSU Sensor Report ..... 185

4.29 Situation Assessment for the RealSecure-MSU Sensor Report ..... 189

4.30 Situation Assessment for the Snort-MSU Sensor Report..... 193

4.31 Situation Assessment for the MultiSensor -MSU Report..... 198

4.32 Situation Assessment for Sensor Corroboration..... 204

TABLE Page

A.1 Comparative Summary of the Literature Review ..... 228

B.1 Categorization of Attacks for RealSecure and Snort Sensors ..... 232

C.1 Criticality Indexes of Source/Target Communication ..... 237

C.2 Criticality Indexes for Attack ..... 237

LIST OF FIGURES


| FIGURE | | Page |
|---|---|---|
| 2.1 | Taxonomies and Alarm Log (Taken from [22]) | 21 |
| 2.2 | A Correlation Chart (Taken from [38]) | 26 |
| 2.3 | An Example HPCN Model (Taken from [63]) | 35 |
| 2.4 | Two FCM Concepts and a Connecting Edge Representing a Causal Link | 42 |
| 2.5 | Individual Alert Generations for Hosts/Users | 43 |
| 2.6 | Combining Evidence of Multiple Suspicious Events | 44 |
| 2.7 | FCM Model to Detect Different Types of Attacks (Taken from [59]) | 46 |
| 3.1 | The Unified Alert Fusion Model | 48 |
| 3.2 | Generalization Hierarchy for Attack Names | 55 |
| 3.3 | Domain Generalization Paths for Alert Features with Similarity Score | 59 |
| 3.4 | A Complete Term Set for the Fuzzy Variable Candidacy Score Superimposed on the Cluster Score Distribution | 63 |
| 3.5 | FCM Model for Combining Evidence of Suspicious Clusters | 64 |
| 3.6 | FCM Model for Detecting DDoS Attack with Sadmind Service Vulnerability | 67 |
| 3.7 | An Abstract FCM Incident Model for Multi-Staged Attacks in General | 70 |
| 3.8 | Combining Evidence of Security Incidents | 77 |
| 3.9 | Dynamic Fusion Process | 79 |

FIGURE Page

3.10 Complete Term Set for Fuzzy Variable Degree of Concern ..... 80

3.11 DHS Threat Model (Taken from [56]) ..... 89

3.12 Resource Concern Model ..... 89

4.1 Service Plot for Lincoln Lab's DARPA 2000 Intrusion Scenario (Taken from [35]) ..... 94

4.2 Alert Reduction with Prioritization for the RealSecure-NCSU Sensor Report ..... 102

4.3 Alert Reduction with Prioritization for the RealSecure-MSU Sensor Report ..... 103

4.4 Alert Reduction with Prioritization for the Snort-MSU Sensor Report ..... 103

4.5 Alert Reduction with Prioritization for the MultiSensor-MSU Report ..... 103

4.6 Correlated Alerts depicting the Attack Scenario for RealSecure-NCSU Sensor Report ..... 108

4.7 Alert Situation for Host mill for the RealSecure-NCSU Sensor Report ..... 114

4.8 Alert Reduction with Abstract Alert Correlation (AAC) for RealSecure-NCSU Sensor Report ..... 117

4.9 Incident Situation for Host mill for the RealSecure-MSU Sensor Report ..... 120

4.10 Alert Reduction with Abstract Alert Correlation (AAC) for RealSecure-MSU Sensor Report ..... 125

4.11 Incident Situation for Host mill from analyzing the Snort-MSU Sensor Report ..... 129

4.12 Incident Situation for Host mill with Missing Alerts from the Snort-MSU Sensor Report ..... 132

4.13 Alert Reduction with Abstract Alert Correlation (AAC) for Snort-MSU Sensor Repor ..... 133

FIGURE Page

4.14 Incident Situation for Host marx: 172.015.114.050 analyzing the MultiSensor-MSU Report ..... 138
4.15 Comparison of IAS Reported for Host marx: 172.015.114.050 for RealSecure-MSU, Snort-MSU and MultiSensor-MSU Reports ..... 139
4.16 Comparison of Incident Situation for Host mill from analyzing the RealSecure-MSU, the Snort-MSU and the MultiSensor-MSU Report for LLDOS 1.0 Inside Zone Dataset ..... 141
4.17 Alert Reduction with Abstract Alert Correlation (AAC) for MultiSensor-MSU Report ..... 141
4.18 Comparison of Cluster Coverage with and without MLC for all the Datasets in the RealSecure-NCSU Sensor Report ..... 146
4.19 Cluster Coverage and Cluster Overall Similarity for Host mill analyzing the RealSecure-NCSU Sensor Report ..... 150
4.20 Cluster Coverage and Cluster Overall Similarity for Host hume analyzing the RealSecure-NCSU Sensor Report ..... 152
4.21 Cluster Coverage and Cluster Overall Similarity for Host plato analyzing the RealSecure-NCSU Sensor Report ..... 153
4.22 Alert Reduction with Multi-Level Clustering (MLC) for RealSecure-NCSU Sensor Report ..... 154
4.23 Comparison of Cluster Coverage with and without MLC for all Datasets in the RealSecure-MSU Sensor Report ..... 155
4.24 Cluster Coverage and Cluster Overall Similarity for Host mill analyzing the RealSecure-MSU Sensor Report ..... 159
4.25 Cluster Coverage and Cluster Overall Similarity for Host robin analyzing RealSecure-MSU Sensor Report ..... 160
4.26 Alert Reduction with Multi-Level Clustering (MLC) for RealSecure-MSU Sensor Report ..... 161
4.27 Comparison of Cluster Coverage with and without MLC for all Datasets in the Snort-MSU Sensor Report ..... 162

FIGURE Page

4.28 Cluster Coverage and Cluster Overall Similarity for Host falcon analyzing Snort-MSU Sensor Report ..... 166

4.29 Cluster Coverage and Cluster Overall Similarity for Host locke analyzing the Snort-MSU Sensor Report ..... 167

4.30 Cluster Coverage and Cluster Overall Similarity for Host mill analyzing the Snort-MSU Sensor Report ..... 169

4.31 Cluster Coverage and Cluster Overall Similarity for Host swallow analyzing the Snort-MSU Sensor Report ..... 170

4.32 Alert Reduction with Multi-Level Clustering (MLC) for Snort-MSU Sensor Report ..... 171

4.33 Comparison of Cluster Coverage with and without MLC for all Datasets in the MultiSensor-MSU Report ..... 172

4.34 Cluster Coverage and Cluster Overall Similarity for Host pascal analyzing the MultiSensor-MSU Report ..... 175

4.35 Cluster Coverage and Cluster Overall Similarity for Host marx analyzing the MultiSensor-MSU Report ..... 177

4.36 Cluster Coverage and Cluster Overall Similarity for Host swan analyzing the MultiSensor-MSU Report ..... 179

4.37 Cluster Coverage and Cluster Overall Similarity for Host mill analyzing the MultiSensor-MSU Report ..... 181

4.38 Alert Reduction with Multi-Level Clustering (MLC) for the MultiSensor-MSU Report ..... 181

4.39 Dynamic Fusion Results for Host pascal for LLDOS 1.0 Inside Zone Dataset analyzing the RealSecure-NCSU Sensor Report ..... 186

4.40 Dynamic Fusion Results for Host goose for LLDOS 2.0.2 DMZ Dataset analyzing the RealSecure-NCSU Sensor Report ..... 187

4.41 Dynamic Fusion Results for Host locke for LLDOS 1.0 DMZ Dataset analyzing the RealSecure-MSU Sensor Report ..... 190

FIGURE Page

4.42 Dynamic Fusion Results for Host mill for LLDOS 2.0.2 DMZ Dataset analyzing the RealSecure-MSU Sensor Report ..... 191

4.43 Dynamic Fusion Results for Host plato for LLDOS 1.0 DMZ Dataset analyzing the Snort-MSU Sensor Report..... 195

4.44 Dynamic Fusion Results for Host pascal for LLDOS 2.0.2 Inside Zone Dataset analyzing the Snort-MSU Sensor Report..... 196

4.45 Dynamic Fusion Results for Host crow for LLDOS 1.0 Inside Zone Dataset analyzing the MultiSensor-MSU Report ..... 200

4.46 Dynamic Fusion Results for Host mill for LLDOS 2.0.2 Inside Zone Dataset analyzing the MultiSensor-MSU Report ..... 201

4.47 Dynamic Fusion Results for Host 11 ..... 205

4.48 Dynamic Fusion Results for Host 5 ..... 206

# CHAPTER I

# INTRODUCTION

This dissertation addresses the problem of intelligent fusion of intrusion detection systems' alerts in a distributed environment using a possibilistic approach with Fuzzy Cognitive Modeling. A unified architecture for intelligent alert fusion is presented that combines multiple tasks for sensor fusion in a single framework for overall assessment of a protected resources' security health. The purpose of this chapter is to outline the research conducted, provide necessary background, present the motivation for the research, postulate the research hypothesis, discuss contributions and clarify terminology used throughout the document.

## 1.1 Information Assurance, Intrusion Detection and Sensor Alert Fusion

Information assurance (IA) remains an active area of research today with significant focus on mechanisms supporting IA. *Information assurance* can be viewed as the perception that systems are operating as required - with expected protection of the availability, confidentiality and integrity of information within systems. In order to maintain trust in systems, mechanisms are deployed that monitor any violation of such perception. As computer technology advances and the threats of computer crime increase, the apprehension and preemption of such infractions become more and more difficult and challenging. In recent years, *intrusion detection systems* (IDS)s have been

extensively used by researchers and practitioners to maintain trustworthiness in systems. An IDS is part of a network defense system that works as a sensor to closely monitor systems for any misuse or anomalous behavior and reports any violation of the designated security policy as alerts to an appropriate authority. With the open nature of the Internet, network intrusion has become a serious problem in recent times and has proliferated the demand for effective network intrusion detection in a distributed environment.

Research in IDSs has taken on new challenges in the last few years. One such contemporary and promising research area that is gaining a considerable amount of interest is the exploration of high-level analysis techniques as a separate layer above the low-level IDS or sensor reports for better trustworthiness in systems. These high-level systems consume alert information from the low-level sensors in order to render advanced conclusions by gathering more intelligence from alert data. The results help to improve the understanding of intrusion behavior and allow security administrators to take appropriate responses. To summarize, sensors analyze data (or observations) and refine them into information by means of pattern matching (in case of misuse sensors) or behavior profiling (in case of anomaly sensors). This information is conveyed to the security administrators in the form of alerts. *Sensor alert fusion* further analyzes the alerts in order to generate higher-level knowledge by associating context to them for better situational awareness.

## 1.2 Sensor Fusion Tasks

Sensor alert fusion is crucial because of some major problems associated with IDS or sensor deployment in a distributed environment, such as follows:

- In a working environment, IDSs overload security administrators with an unmanageable volume of alerts.
- IDSs can be very noisy. Approximately, 99% of IDS alerts maybe false [22].
- IDSs analyze data to generate alerts independently without adding any context or significance to them.
- IDSs report only on the isolated effects of an intrusion rather than the interactive effects of an attackers' coordinated actions.
- IDSs cannot shed light on the global view behind the attacks (e.g. attacker's mission, plan, strategy, etc.)

Although the main objective of sensor alert fusion is to understand data better for stronger assurance, sensor fusion systems primarily differ by their targeted goals. The following discusses the primary sensor fusion tasks in terms of what they aim to achieve:

- *Alert Prioritization*: Assesses the significance of sensor alerts in accordance with a designated security policy. Alert prioritization helps to filter out non-significant alerts and reduce the alert volume.
- *Alert Clustering*: Discovers structural relationships in data by grouping/aggregating alerts with common features in some meaningful groups/clusters. Alert clustering aids in discovery of common attack patterns.
- *Alert Correlation*: Discovers causal relationships in data by associating alerts that are parts of a single chain of events. Alert correlation helps to identify multi-staged attacks from chains of attackers' actions.

Researchers have used different technical approaches to carry out these different fusion tasks. Chapter II reviews some of the approaches in detail.

## 1.3 Multi-Sensor Environment

In response to proliferated attacks on enterprise systems today, many practitioners employ multiple, diverse sensors for increased information assurance because a single sensor cannot detect all types of attacks. A *multi-sensor environment* is characterized by deployment of a homogeneous and/or heterogeneous suite of sensors to monitor different entities in the corresponding environment. In multi-sensor environments, the sensors can collaborate with or complement each other to provide increased assurance of information. These multiple sensors may employ different strategies based on the model they use, the data source they monitor and the techniques they employ. This is especially true of systems, for example, that may be geographically distributed over a wide area or those that employ newer architectures like high performance clusters. In a multi-sensor intrusion detection environment, the following often characterize the sensors:

- data source (host-based/network-based);
- type of intrusion detection (anomaly detection/misuse detection);
- implementation (hardware/software/firmware); and
- technique used (statistical/artificial intelligence).

Essentially, the primary advantage of using multiple sensors is to improve the detection rate and the coverage within the system - whether the sensors used are homogeneous (i.e., multiple installations of the same types of sensors allowing monitoring from different points) or heterogeneous (i.e., multiple installations of different types of sensors allowing different perspectives from the same/different points). While a homogenous suite of sensors working at different points in the network provides concrete

evidence of widespread attacks, heterogeneous sensors working at a single point can provide more insight into suspicious activities in the network [16]. Different types of sensors are able to capture different aspects of security violations in different areas. Therefore, it is preferable to use different types of sensors or combinations of sensors in different circumstances and surroundings. It is easy to see how having multiple sensors, which are able to corroborate/complement/challenge each other's findings, can enhance confidence in the assurance of systems. For example, while a network-based sensor can monitor traffic to from and within the network, a host level sensor can monitor unusual changes at the operating system level of hosts. Attacks, like IP spoofing, can be captured only by network-based sensors such as Snort[1], while attacks such as unauthorized file changes can be captured only by host-based sensors such as Tripwire[2]. While misuse sensors have the advantage of detection specificity and the disadvantage of not being able to identify unknown violations, anomaly sensors have the advantage of being able to deal with novel situations and the disadvantage of prompting false alerts. Typically, anomaly sensors are able to provide earlier warnings than misuse sensors because anomaly sensors can report any anomalies in progress while misuse sensors need to wait for an entire attack signature before sending warnings [15]. Moreover, while misuse sensors can either report an alert with full confidence or cannot report at all, anomaly sensors typically can associate some confidence factor with the alerts [57].

---

[1] Snort: An open source cross-platform, lightweight network intrusion detection tool developed by Martin Roesch [47]. For details please refer to http://www.snort.org.

[2] Tripwire: Host based change monitoring and reporting system developed by Tripwire Inc. For details please refer to http://www.tripwire.com.

As there is no "perfect" or "one for all" sensor, it is only natural to employ a suite of different sensors to maximize trustworthiness in systems such that an inability and/or weakness of one is compensated by capability and/or strength of another. However, as discussed in the next section, managing alerts in a multi-sensor environment is an extremely difficult task requiring special consideration.

## 1.4 Motivation

As mentioned in section 1.2, the different types of sensor fusion tasks can be divided into three broad categories: one which analyzes alerts to filter out false positives [22, 44]; one which analyzes alerts to discover common patterns of attack [8, 9, 57]; and one which analyzes alerts to discover multi-staged attacks or to predict attackers' plan [7, 33, 36, 37, 45]. Although these primary sensor fusion tasks have their own merits, collectively they can provide more useful information than each of them can individually. Alert prioritization is necessary so that non-significant alerts that are mostly false positives do not overwhelm the security administrator and any further analysis can focus on more important or critical threats. However, alert prioritization cannot identify any associations in data like alert clustering or alert correlation can. Alert clustering is needed to add context to alerts by seeking to organize alerts in groups such that alerts within a group are identical. However, it cannot identify any causal associations in data like alert correlation can. Alert correlation is especially useful for detection of coordinated attacks with a common goal that span over time. However, it cannot discover structural similarities in data like alert clustering can. A few researchers have used some combinations of these tasks with special objectives. For example, Ning et al. combine

alert clustering with alert correlation to hypothesize about attacks missed by sensors [40]. Cuppens applies alert correlation on the results obtained after alert clustering [7] for intrusion plan recognition. Yu and Frincke conduct alert clustering and alert correlation at the same time to assess probabilities of security incidents [63]. Until now, there has not been any effort to employ alert prioritization, alert clustering and alert correlation in a single framework in a manner such that their individual results are used to quantify a confidence score as an overall assessment for global diagnosis of a system's security health.

As discussed in section 1.3, it makes good engineering sense to employ multiple sensors in a secure environment. However, managing data from these sensors is critically important as there are disadvantages associated with multi-sensor systems:

- When multiple sensors require individual monitoring, the workload for the security administrator is increased in many fold.
- In situations when there are large numbers of alerts reported, the alert volume from different sensors can overwhelm the security administrator and make analysis of such alerts extremely difficult.

While sensor diversity contributes to more coverage of attack space [1], it makes analysis of diverse data more difficult. Fusion of the sensor alerts is crucial to provide sophisticated reasoning capabilities outside the sensors' core functions. Potential advantages of sensor alert fusion in a single or multi-sensor environment are as follows:

- Elimination or reduction of the need for manual analysis of reported data;
- Compression or reduction of alert volume by combining similar alerts;
- Identification of context by associating alerts from different sensors;
- Improvement in detection rate by sensor reinforcement [1];

- Improvement in diagnostic ability by identifying category, significance, relevance, priority, phase, result of failure/attack;
- Reduction of false alerts by sensor corroboration;
- Introduction of perspective by providing different views of the same incident [1];
- Provision of scalability to deal with large volumes of data.

To this date, there has been some significant work done in the area of sensor alert fusion. Among them we find that some approaches assess relevance of alerts and discard alerts that are considered non-malicious [22, 44]. Some approaches use similarity between alert features to group/aggregate/cluster alerts [24, 57, 58]. All these approaches employ subjective parameters defined by human experts to address similarity. Some approaches use predefined correlation scenarios to correlate alerts to discover what has happened and what might happen [8, 33, 36]. Again human experts are directly involved in encoding knowledge into scenarios. Some approaches learn correlation models by applying machine learning techniques [9] or statistical techniques [45] but require extensive training data with known scenarios to succeed. Others use predefined prerequisite and consequences information about attacks to correlate alerts, where human experts encode the prerequisite and consequences information for possible attacks into a knowledge base [7, 37]. A very few approaches analyze data from other information sources apart from IDSs [15]. All of these approaches have different strengths and limitations, where no single one clearly dominates the others in all aspects. Also, we see a lack of solutions in this area using possibilistic approaches (selected research in this area is discussed in Chapter II based on the technical solutions they undertake to solve their respective goals). In this dissertation, we use a possibilistic approach with Fuzzy

Cognitive Modeling for intelligent sensor alert fusion that strives to retain some advantages of previous work in this area, while addressing some of their disadvantages. We believe that the problem of sensor alert fusion to assess an overall security status is well suited to be addressed with possibilistic approaches. The state of a system's security status is a vague concept versus a binary one with different extents of truth in it that should be consistent with evidence support.

## 1.5 Design Goals

In light of the above discussion, it is easy to see how important it is to fuse the different outputs of sensors to conduct primary sensor fusion tasks and combine the results of these different tasks in an effective and intelligent manner in order to provide the security administrator with an overall, condensed view. The high-level view would serve as an aid to maximize trust in a system and reduce information overload for the security administrator. In this respect, we address the development of a unified architecture for sensor alert fusion that combines several aspects of sensor alert fusion in an integrated manner. The following research issues concerning alert prioritization, alert correlation and alert clustering are explored as primary design goals for our work in this dissertation:

*Integration of effects of different factors in modeling alert prioritization:*

Alert prioritization involves weighting or assigning priorities to low-level alerts. All reported alerts should not be given the same importance because the significance of an alert depends heavily on the information it contains and also on the source it is coming from. The criticality of information that alerts contain is mainly dependent on a site's security policy. For example in an e-business environment, an alert that identifies a web-server as a target is more critical that an alert that identifies an ftp server as a target. This dissertation investigates the modeling of alert prioritization taking into account these different factors and their importance in a given context.

*Modeling of "inexact" matching in alert clustering:*

Alerts are assumed to belong to the same cluster/group if they have common attribute values for different features like source, target, time, attack, service, and user. Often, in real situations, the notion of similarity is not clear-cut but involves a certain degree of likelihood. This dissertation addresses alert clustering using Fuzzy Cognitive Modeling based not only on exact matches of attribute values but also close or inexact matches of attribute values.

*Modeling of causality in alert correlation addressing scalability:*

Alert correlation involves finding causal relationships between alerts. Often in real situations alerts result because of multi-staged attacks where the earlier attacks set the stage for the later ones [37]. This dissertation uses Fuzzy Cognitive Modeling to model and reason with causal relationships between alerts. Different attacks can cause different effects in systems, which can be tied together in a causal chain to expose

possible correlations between them. Also, sensor fusion can become impractical as the number and type of sensors increase, as the size of the network increases, and as a wider variety of alerts are considered. An approach for addressing this scalability issue is to make the fusion models as general as possible. Julisch [22] uses attribute-oriented generalization of alerts based on alert attributes such as, source, destination, and time to identify root causes of alerts and reduce the alert volume. This dissertation uses a similar approach for generalizing alert attributes and modeling abstract scenarios.

*Integrating results of different fusion tasks together:*

For improved understanding of a system's security health, the results of the primary sensor fusion tasks need to be combined in an effective and integrated manner. This dissertation uses possibilistic information combination techniques for overall situation assessment. For misuse sensors, where alert association can be conducted for higher-level reasoning, this dissertation conducts decision level fusion. For anomaly sensors without alert association, this dissertation conducts sensor corroboration with data level fusion. This concept is useful for a resource restrained high performance computing cluster environment that typically employs anomaly sensors.

## 1.6 Hypothesis

The hypothesis of this dissertation is that a *unified alert fusion model based on a possibilistic approach with fuzzy cognitive modeling can be used for high-level analysis of sensor alerts in an intrusion detection environment to provide a security administrator with an overall situation assessment of protected systems' security health to aid the security administrator in decision making.*

## 1.7 Contribution

The primary contribution of this dissertation is the development of a unified alert fusion model that combines primary sensor fusion tasks in a single, cohesive, coordinated framework for overall security situation assessment of network resources in a distributed environment. The main contributions of this work can be specifically described as follows:

- Development of:
    - a new alert prioritization technique to filter out lower priority alerts in order to reduce alert volume.
    - a new alert clustering technique that uses fuzzy cognitive modeling with generalization to cluster alerts with the same and similar features in order to identify common attack patterns.
    - a new alert correlation technique that uses fuzzy cognitive modeling with generalization to correlate alerts linked in multi-staged attacks. Such alert correlation can deal with scalability issues and can correlate alerts even though intermediate alerts are missing in sensor reports.
    - a new concept of situation assessment that derives quantitative assessment of systems' security health using a possibilistic approach.
    - a new family of dynamic fusion approaches for situation assessment where the results of alert clustering and alert correlation are combined for misuse sensors and reports of event-based evidence and state-based evidence are combined for anomaly sensors.
    - a new taxonomy for categorizing attacks based on the possible impacts of probable attacks.
- Empirical evaluation of sensor fusion tasks:
    - this research presents the first reported detailed empirical evaluation of multiple sensor fusion tasks conducted on multiple independent and integrated sensor alert reports generated on a well known benchmark intrusion detection evaluation attack dataset.

This dissertation addresses a timely and significant research problem with a promising new approach. It contributes to the following areas of research:

- *Information assurance* by proposing a new application of a well known soft computing tool, i.e., Fuzzy Cognitive Modeling;
- *Intrusion detection* by proposing a new unified alert fusion model for overall situational awareness;
- *Applications of Fuzzy Cognitive Modeling* by proposing its use in a new application domain.

## 1.8 Terms

As an attempt to familiarize the reader with the subject matter, this section highlights some elementary concepts that are closely related to this dissertation. This is because, as will be seen in Chapter II (where related research in this area are reviewed), the research community uses these terms ambiguously which may cause confusion for the general reader.

Table 1.1 Terms Related to the Research

| TERM | DEFINITION |
|---|---|
| EVENT | – Manifestation of a fault or activity of concern. |
| ALERT/ALARM | – Message from a sensor to signal the occurrence of a malicious event or suspicious irregularity in normal events. |
| ATTACK | – Action or series of actions to achieve an unauthorized goal [18]. |
| SECURITY VIOLATION/ INTRUSION | – Attempts to access protected resources/information in a way that affects the expected performance of the system and/or challenge the security policy of the system. |
| INCIDENT | – Indicates an explicit or implicit security infringement event by the occurrence of attack(s). |
| ALERT FUSION | – Fusion of content from alerts as reported by various sensors into "meta-alerts" [57].<br>– Process of determining a quantitative value for the system such that the value is representative of the degree of concern in the system [60]. |
| ALERT FUSION SYSTEMS | – Combining alerts from multiple heterogeneous systems and making available the results to the security administrator at a single easily monitored location [8]. |
| ALERT PRIORITIZATION | – Ability to weight/assign priorities to low-level alerts [16]. |
| ALERT THREADING | – Organizing alerts from a single sensor that are part of an ongoing attack [1]. |
| ALERT ASSOCIATION | – Analysis of different types of relationships in data such as, causal and structural types of relationship. |
| ALERT CLUSTERING/ AGGREGATION | – Analysis of structural relationships in data. |
| ALERT MERGING | – Replacing a cluster/group of low-level alerts with a high-level alert that is representative of the group. |
| ALERT CORRELATION | – Analysis of causal relationships in data. |

## 1.9 Organization

The remainder of this dissertation is organized as follows. Chapter II is an extensive review of selected research in sensor alert fusion, including our previous work in this area. Chapter III introduces the unified alert fusion model with the proposed techniques. Chapter IV details the experiment design, setup, results and analysis of the results that demonstrate the effectiveness of the research. Finally, Chapter V states the conclusions of this research and presents future work.

# CHAPTER II

# LITERATURE REVIEW

Research in the area of sensor alert fusion has only begun to emerge in the last few years. In this chapter, we review selected work that we deem relevant to our research, including our previous work on decision support in the network security area.

## 2.1 Sensor Fusion Approaches

Due to the lack of widely accepted standards, the research community involved in sensor alert fusion ambiguously uses different relevant terms in this area of research. Therefore, it is hard to categorize the research efforts based on the problem they solve. What is defined as correlation for one is defined as aggregation or fusion or clustering for another. However, as the primary objective of the research in this area is the same, i.e., to provide higher-level reasoning abilities beyond low-level sensor capabilities, we categorize and review the research based on the solution approach the researchers have undertaken to solve their problem.

Multi-sensor alert fusion research for information assurance mostly concerns information modeling and high-level reasoning. Most of the research employ some variation of expert systems to conduct this reasoning. In this dissertation we have selected to review research based on expert systems, since ours also falls into this category.

"Expert systems are a highly domain-specific type of knowledge based systems used for a specialized purpose" [19]. An *expert system* is defined as an artificial intelligence application that contains a knowledge base of expert information and a set of algorithms or rules, with which it infers or reasons about new facts/decisions from incoming data to aid in problem solving/decision making [19]. The reasoning techniques primarily fall into two categories:

- *Reasoning under Certainty*: In this approach all types of high-level reasoning or decision making are carried out using logical inferences built on experts' knowledge. This approach is effective when problems are deemed to be exact and certain.
- *Reasoning under Uncertainty*: In this approach all types of high-level reasoning or decision-making are carried out using inferences that accommodate uncertainty, incompleteness or impreciseness of information. Reasoning under uncertainty is primarily based on the *theory of probability*, which deals with the notion of uncertainty by measuring the degree of belief or likelihood and also on the *theory of possibility*, which deals with the notion of uncertainty by measuring the degree of truth or ease [62]. This approach is effective when issues are uncertain, vague or incomplete.

Most problems in the real world are not exact and certain and have some form of uncertainty issues involved. Although both probabilistic and possibilistic approaches provide us with tools to reason with uncertainty, the main difference between them is that probabilistic approaches deal with a body of precise and varied knowledge while possibilistic approaches deal with vague but coherent knowledge [46]. In the following

sections, we review security research based on these categories and we find our place among research employing possibilistic approaches in reasoning under uncertainty where we see a clear lack of research and also discuss our previous work [48] as a first attempt in this area. Please refer to Appendix A for a comparative summary of the research reviewed.

## 2.2 Selected Research based on Reasoning under Certainty for Alert Fusion

### *2.2.1 Aggregation and Correlation of Alerts*

In multi-sensor alert correlation, one of the early research efforts was led by Debar and Wespi [10]. This work concentrates on alert correlation more in terms of discovering structural relationships between alerts. The authors introduced the concept of an *Aggregation and Correlation Component (ACC)* that can analyze and correlate alerts generated by intrusion detection probes/sensors using an expert rule-based system. The tasks carried out by the ACCs include the following: alert acquisition, alert accumulation, alert analysis, diagnosis, report emission, and report generation. ACCs collect the alert data from various probes or sensors over time and may feed ACCs at a higher-level in the hierarchy. The goal of the ACCs is to generate one alert per attack, even if the attack generates multiple alerts [10]. ACCs assign confidence values with each alert by considering the intrinsic inaccuracy of the probe in question, i.e., the inaccuracy for which the probe can be held accountable, and the relative inaccuracy of the system being monitored, i.e., the inaccuracy due to the system's unusual but non-malicious behavior. ACCs also take into account the severity of alerts during the assessment.

ACCs essentially group the alerts together to provide multi-level views of the alerts that highlight their importance. This is accomplished by analyzing the alerts to identify *duplicates* and *consequences* and then grouping them to form *situations*. In this respect, ACCs look for two types of relationships between alerts [10]:

- *Correlation Relationship*: Alerts in this relationship are considered part of the "same trend of attacks" and are either identified as duplicates or consequences. Identification of *duplicates* involves crosschecking alerts between two different sensors with common information such as, source address, source port, target address and target port - within a specific time frame. This crosschecking is done according to the "Duplicate Definition" of alert class type in question. Duplicate definitions of different alert classes are different due to the different nature of the class. Therefore, duplicate definitions can have severity levels that are both positive and negative. The count of duplicates can affect the severity level for certain alert classes such as, ping of death or teardrop attacks. A *consequence* set is a set of alerts linked together within a specific time frame where each alert in the alert chain is considered a consequence of a previous alert. "Consequence Definitions" are also based on nature of alert classes.

- *Aggregation Relationship*: Alerts in this type of relationship are grouped together based on some criterion to form situations. All alerts in a situation have some characteristics in common. Three aggregation axes are used for this purpose: source, target and class of attack. With these three aggregation axes, seven different situations can be aggregated. A more specific situation is given greater importance than less specific ones. Situation severity is calculated as the sum of the severity of the individual alerts in the situation. There are two ways to deal with situation concerns. In situation re-evaluation, the human operator can change the severity level of the situation. In multi-situation assessment, the severity level can be changed if related situations exit.

In Debar and Wespi's work, experiments were conducted with a heterogeneous suite of sensors which included: ISS's RealSecure, Cisco's Secure IDs, Web IDS, TCP Wrapper and Klaxon [10]. The authors did not report on the procedure or the results of specific experiments.

As a usage example, the graphical user interface for the alarm view was shown for CGI types of attacks. The authors concluded that the ability of the ACCs to automatically gather more information, to modify setup of the probes and to send warnings to appropriate authority are works in progress.

### *2.2.2 Alert Clustering with Abstraction*

Julisch introduces alarm (i.e., alert) clustering as a method to support *root cause discovery* [22]. The root cause of an alarm is defined as the "reason for which it occurs." The author argues that root causes are primarily responsible for the large number of redundant alarms and 90% of these root causes are generated because of configuration problems and thus are fixable with manual interception. This work outlines a semi-automatic approach for reducing false positives in alarms by identifying the root causes automatically and then removing them manually. Such measures can drastically reduce future alarm load [22]. In this work, alarm clustering is performed by grouping together alarms whose root causes are generally similar. A generalized alarm for a specific alarm cluster represents a pattern that all of the alerts in the cluster must match in order to belong to that cluster.

The author uses taxonomies to define similarity between alarms. Taxonomies are generalization hierarchies built upon the alarm attributes (Figure 2.1). Alarms are clustered based on "dissipation," a measure of the average distance between alarms in an alarm set and their cover (i.e., the most specific alarm to which all the alarms in the set can be generalized). For the taxonomies shown in Figure 2.1, the cover of the alarm set

{(ip1,80), (ip4,21)} would be (DMZ,PRIV) and the corresponding dissipation would be 3 [22]. Covers can be used to represent a particular alarm cluster.

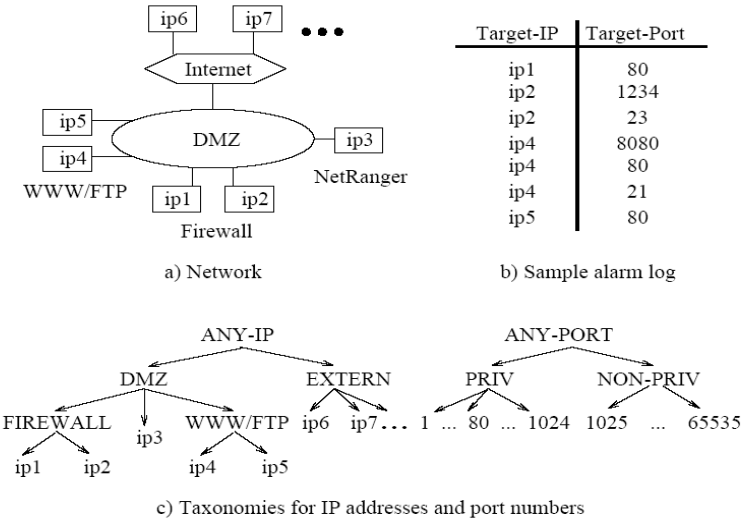


Figure 2.1 Taxonomies and Alarm Log (Taken from [22])

The alarm clustering algorithm, based on an attribute-oriented induction method, attempts to find alarm clusters where all the alarms share the same root cause [22]. This is accomplished by grouping alarms with minimum acceptable dissipation. The alarm clustering approach was evaluated offline with an alarm log generated by a commercial sensor situated in a commercial network spanning a time period of a month and containing 156,380 alarms. The 13 largest alarm clusters generated by the clustering algorithm covered 95% of all alarms [22]. By analyzing the clusters manually, 6 different root causes were identified to attribute to 95% of all alarms.

When a filter was written to remove the root cause behind one of the largest identified alarm clusters, 82% of the original alarms were automatically discarded by the filter. This work outlines an effective approach to reduce false positives in sensor alert reports by clustering alerts with abstraction and then using the clusters to discover and understand the root causes of alerts.

### *2.2.3 Co-operative Intrusion Detection*

In the French Defense Agency's MIRADOR project, the main objective is to develop a co-operation module between multiple intrusion detection systems (IDSs) to correlate alerts in order to reduce the alert volume and generate more global and synthetic alerts. The co-operation module is an expert rule-based system that supports logical reasoning with predicate logic. The functions of the system include the following [7]:

- *Alert Management*: alerts issued by different IDSs are stored and managed in a relational database.
- *Alert Clustering*: alerts from the same or different sensors that indicate the same occurrence of an attack are collected in clusters.
- *Alert Merging*: clustered alerts are merged into a global alert that represents the alert cluster.
- *Alert Correlation*: alerts are correlated to recognize intrusion plans.
- *Intention Recognition*: intrusion plans are extrapolated to identify an intruder's intention.

In this system, alert clustering refers to finding similarity of new alerts to existing alerts in a knowledge repository. The alert similarity is determined by the similarity requirements specified by expert rules [7]. These rules are domain specific and are defined by examination of prior alerts generated by the IDSs considered. The rules are

defined to express similarity between alert source, target, time and classification. For classification similarity, a mapping between typical attack names, different IDSs and their generated names for the attacks, are used. This is because different IDSs may refer to the same attack with different names. For time similarity, a threshold or delay - beyond which alerts are not considered for clustering, is used. For source and target similarity, mappings between host names and IP addresses and between services and port numbers are considered. Also, specific expert rules define unique cases where sources/targets can be considered similar based on alert classification.

For experimentation, two network-based IDSs, Snort and e-Trust, were used [7]. Both of these sensors are signature-based sensors. The sensors were tested with an attack base of 87 elementary attacks. Elementary attacks correspond to non-decomposable steps of a given scenario. In response to the attacks, the sensors generated 325 alerts. With alert correlation, 101 alert clusters were identified by the system where it was expected to identify 95. The largest cluster contained 102 alerts. The authors concluded that it is crucial to map different attack names generated by different sensors with extreme care. The authors also suggested that making the time delay requirement for time similarity depend on the nature of the alert would improve the results.

In the MIRADOR project, semi-explicit alert correlation [7] is used. In this type of correlation expert defined correlation rules are used to recognize situations such as whether executing an attack contributes to executing another attack. This work is heavily dependent upon the specification of attacks based on an attack language called LAMBDA. LAMBDA specifies attacks by precondition, post conditions, attack scenario,

detection scenario and verification scenario. The attack description is used to derive two kinds of correlation links automatically [8] that can be used to build an offline correlation base. The correlation rules of the correlation base can later be used to correlate alerts and build intrusion plans.

The authors illustrated this approach by correlating a multi-step attack scenario with the Illegal NFS Mount scenario. When this attack was launched, Snort generated 7 alerts and e-trust generated 2 alerts. The clustering function produced 5 clusters and the correlation function correlated the steps into one attack scenario.

### *2.2.4 Prerequisite - Consequence Model of Alert Correlation*

Ning et al. [38] proposes an alert correlation model based on the inherent observation that most intrusions are related as in different stages of an attack sequence, where "early stages prepare for the later ones." The correlation model is built upon two aspects of intrusions:

- *Prerequisites*: Necessary conditions for an intrusion to be successful. For example existence of a vulnerable *sadmind*[3] service running on a host is the prerequisite before a buffer overflow attack exploiting a vulnerability of the *sadmind* service can take place;
- *Consequences*: Possible outcome of an intrusion. For example gaining of root access by attacker after a buffer overflow attack exploiting a vulnerability of the *sadmind* service takes place.

[3] Sadmind is an "installed by default" service in Sun Microsystems operating system and is designed to provide remote system administration operations http://www.sans.org/resources/malwarefaq/sadmind_iis.php.

With knowledge of prerequisites and consequences, the correlation model can correlate related alerts by finding causal relationships between them, i.e., by matching the consequences of previous alerts with prerequisites of later ones [38]. For example, once the consequence of a *sadmind ping* alert is found to match with the prerequisite of the *sadmind buffer overflow* alert, the two alerts will be correlated by the model.

In the prerequisite-consequence model, the authors conduct reasoning with predicate logic where predicates are used as basic constructs to represent the prerequisites and consequences of attack. The construct *hyper alert types* encode all knowledge about a particular attack with the following triple: fact (which identifies the attributes associated with this type of attacks), prerequisite and consequence. An example of a hyper alert type triple for $T_{sadmindBOF}$ (sadmind buffer overflow attack) is:

1. fact={VictimIP, VictimPort};
2. prerequisite={ExistHost(VictimIP) AND VulnerableSadmind (VictimIP)}; and
3. consequence={GainRootAccess(VictimIP)}.

A *hyper alert* denotes instances of hyper alert types [38]. One hyper alert can designate one or more related alerts. An example of a hyper alert is $h_{sadmindBOF}$, which is an instance of the SadmindBufferOverflow hyper alert type $T_{sadmindBOF}$, and consists of the following records: {(VictimIP=152.141.129.5, VictimPort=1235), (VictimIP=152.141.129.37, VictimPort=1235)}. As a result of prerequisite-consequence matching, the correlation model identifies a buffer overflow attack against the sadmind service for the above mentioned IP addresses and predicts an attacker gaining root access at those hosts.

Once the correlation model is able to identify time-lined sequences of such hyper alert instances, it can present a correlation chain where earlier alerts are shown to prepare for the later ones. Therefore, if a sequence of hyper alerts $h_{IPSweep}$, $h_{sadmindPing}$, $h_{sadmindBOF}$, $h_{DDOSDaemon}$ is found, with prerequisite-consequence matching, the following hyper alerts will be correlated as: $h_{IPSweep}$ prepares for $h_{sadmindPing}$, $h_{sadmindPing}$ prepares for $h_{sadmindBOF}$, and finally $h_{sadmindBOF}$ prepares for $h_{DDOSDaemon}$.

Ning et al. used *hyper alert correlation* charts to visually represent the alerts [38]. A hyper alert correlation chart is a connected chart where each node represents a hyper alert and the edges connect two hyper alerts if one prepares for the other one. Figure 2.2 shows a hyper alert correlation chart for the above example with the current point of interest being the shaded Sadmind BufferOverflow (BOF) alert.

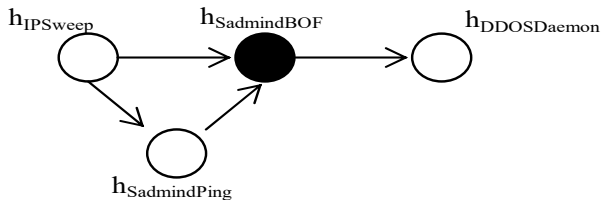


Figure 2.2 A Correlation Chart (Taken from [38])

Ning et al. [38] claim that the prerequisite-consequence model reveals structures of series of attacks, reduces false alerts, and predicts attacks in progress. To evaluate the effectiveness of their approach for alert correlation, the authors performed a set of experiments with the DARPA 2000 intrusion detection evaluation datasets, as found in [34]. The authors detailed performance of their approach as compared with an intrusion detection sensor RealSecure, in terms of detection rates and false alert rates and reported

that, although the detection rates were more or less comparable, their approach reduced the false alert rates far more significantly than RealSecure.

The limitation of the above approach is that it is heavily dependent on prior knowledge of modeling each attack at the prerequisite and consequence level and therefore does not detect unknown attacks or variations of known attacks. In [40], Ning et al. integrate the prerequisite-consequence model of alert correlation with alert clustering based on a match of alert attribute values. In this respect, the authors correlate alerts to generate correlation charts separately and then integrate two correlation charts together if the charts involve the same destination IP address. The authors reason about missed attacks with the assumption that subsequent attacks in a correlation chain can be considered *directly related* and when any critical attack in the correlation chain is missing, the attacks in the chain are considered *indirectly related*. The authors define and use pre-defined constraints that must be satisfied by attacks to be considered as indirectly related.

### *2.2.5 Alert Fusion Framework for Scenario Recognition*

In this research, Mathew et al. present a framework for attack scenario representation that allows real-time fusion of intrusion alerts to detect, predict and reason about multi-staged goal oriented attack scenarios [33]. The framework uses graph-based modeling of attack scenario templates for scenario construction. The Scenario Graph for IDS alert Fusion (SGIF) framework for scenario recognition uses IDS alerts as atomic elements and a hierarchical structure to relate attacks to exploits. The SGIF elements are briefly described below [33]:

- *Attribute Node*: Also known as alert node; represents events with a collection of attribute fields, some of which are designated as critical. The values of the critical field distinguish between different attribute nodes. Each attribute node has a binary credibility value that contributes certain weight to its parent Attack Node(s).

- *Attack Node*: Represents exploits and is the parent node of attribute nodes. Uses correlation function such as Weighted Average or Max to define how the child attribute nodes are correlated. Attack nodes are associated with dynamic attack node credibility values that determine how much one attack node contributes to its parent node, which may be another attack node or a node in higher-level scenario graph.

- *Scenario Graph*: The nodes in a scenario graph represent different stages of a multi-staged attack, with at least one node being a goal node. A scenario graph also has a credibility value that varies dynamically as fusion of live alert stream continues. The value indicates the extent of a multi-staged attack.

The alert fusion engine uses a database of scenario graph templates that is built based on the knowledge of known multi-staged attacks. Once alerts are processed, the attack nodes are assessed credibility values depending on the critical fields of the attribute nodes, with the help of the correlation function. The choice of the function depends on the type of the attacks. Then, scenarios are constructed with graph matching against the predefined scenario templates and credibility values of matched scenarios are calculated. Alerts that contribute to nodes in a scenario graph are correlated. The dynamic credibility values of scenario graphs represent real-time situation awareness regarding attack scenario development.

For preliminary evaluation of their work, the authors used a Snort-based stream of alerts captured from a network within a research environment to show how scenario construction is done and how the dynamic credibility value of a scenario graph represents the progress of a multi-staged attack [33]. Along with identifying scenarios, the scenario

graph approach also allows quantitative assessment of progress in a multi-staged attack. Because predefined scenarios are used, unknown attack scenarios cannot be detected. However, this research shows some potential for real-time fusion of intrusion alert streams.

### *2.2.6 Hybrid Intrusion Data Fusion*

Ye and Xu [61] discusses fusion of sensor data as the process of determining a quantitative value for the system level such that the value is representative of the degree of concern in the system.

The authors use three different information fusion methods to generate a composite output value by fusing inputs from three intrusion detection techniques [61]: a signature-based technique based on a decision tree and two anomaly detection techniques based on statistical chi-square tests and an exponentially weighted moving average (EWMA).

As the first fusion method, an artificial neural network is used to learn the non-linear function between fusion inputs and the output from training data. The neural network is based on a multiplayer perception model with a back propagation learning algorithm. Linear regression techniques are used as the second fusion method to learn a linear function between the inputs and the output. Lastly, logistic regression is used to build a special non-linear function between the inputs and the output. The authors report

the use of linear regression as the most effective and least computationally expensive fusion technique for network intrusion detection [61]. The authors conclude that the neural network approach did not produce good results because of overfitting of training data.

## 2.3 Selected Research based on Reasoning under Uncertainty for Alert Fusion

### *2.3.1 Probabilistic Approach*

#### *2.3.1.1 Alert Correlation based on Probabilistic Theory*

Probabilistic alert correlation finds similarity between alerts that match closely, if not exactly [58]. According to Valdes and Skinner, probabilistic alert correlation correlates attacks over time, over multiple attempts and from multiple sensors. The alert correlation task consists of the following [57, 58]:

- *Identifying alert threads*: if a sensor identifies an alert to be an update of/related to one of its previous alerts, it is considered to belong to the same alert thread.
- *Identifying incidents by clustering/correlating threaded alerts with meta-alerts:* if an alert is not considered part of an alert thread, then it is correlated with a list of meta- alerts by means of feature similarity.
- *Clustering/correlating meta-alerts with meta-alerts:* meta-alerts can be grouped together with other meta alerts to identify attack scenarios.

In this correlation scheme, a repository of meta-alerts constructed using expert knowledge and prior alerts from heterogeneous sensors is maintained for similarity matching with alerts reported [58]. The clustering/fusion scheme uses similarity functions to measure the closeness of each feature pair and in doing so, considers only the overlapping features, i.e., the features that are common between the alert and the meta-

alert in question. Typical overlapping features are: source and targets of attacks, the class of attack and the time of attack. Features that are not common between alerts do not contribute to the overall similarity match because heterogeneous sensors do not generate alerts with all possible identifying features. For example, while a network-based sensor cannot generate process ids, host-based sensors do [58]. Construction of similarity functions to measure feature similarity is based on combination of expert rule-base and Bayes formalism. Finally the overall similarly between alert and meta-alert is calculated as the sum of the weighted average of each feature similarity:

$$SIM(X,Y) = \frac{\sum_j E_j SIM(X_j, Y_j)}{\sum_j E_j}$$

X=Candidate meta alert
Y=New alert
j= Index over alert features
$E_j$ =Expectation of similarity of feature i
$X_j$ , $Y_j$ = Values of features j in X,Y

Valdes and Skinner introduce the concept of *expectation of similarity* that serve as the normalizing weight of the similarity functions and the concept of *minimal similarity* that serves as the threshold for consideration of similarity. The notion of a situation specific expectation of similarity is an interesting idea as it helps to express prior expectations such that features between two alerts can only match if the alerts are related. For example, in the case of probes from same source, the expectation of similarity of matching target IPs is low [58]. Also, the notion of a minimum match criterion prevents correlation based on spurious matches for less important features. If any feature pair fails to match the minimum criterion of similarity, the alert is excluded from consideration of overall similarity. Even if all feature pairs pass the criterion, the overall similarity still has

to pass the minimum match threshold in order to be grouped with the most similar meta-alert. If an alert fails to match with any meta-alert in the list, it itself becomes a meta-alert to be considered for future alert correlation.

In order to determine similarity between attack classes, a matrix of similarity is used that denotes experts' judgment of similarity between known attack classes. The authors consider attacks classes rather than attack signatures because attack signature names can be very specific and therefore differ from sensor to sensor. The class similarity is based on the notion of proximity. The proximity of class A to class B is the likelihood of attack type A progressing to attack type B. The authors denote that such proximity similarity allows one to correlate multi-staged attacks [58].

Valdes and Skinner [58] describe techniques used for clustering/correlating alerts with meta-alerts, but do not mention the mechanism for identifying alert threads within sensors and clustering/correlating meta-alerts with other meta-alerts.

The authors report their results with live data collected over a period of three weeks for a progressing stealthy port scan [58]. The IDS sensors used were Emerald eBayes, expert-Net and ISS RealSecure. Among the sensors, the TCP misuse monitor and the asset availability monitor employ techniques based on Bayes inference and are aware of each other's state [58]. While the IDS sensors generated 4439 alerts in the specified time period, the probabilistic correlation system produced 604 correlated alerts, contributing to a reduction of one-half to two-thirds in alert volume in the live environment. The experiments were conducted in a controlled environment with specific attack scripts executing an MSCAN probe, CGI exploit, and buffer overflow attacks. The authors

demonstrated, with the Emerald's monitor display, the aggregator was able to identify incidents and attack scenarios in the experiments conducted and contributed to a 50% reduction in alert volume in the simulated environment. The "probabilistic alert correlation scheme" is targeted for correlation of alerts over extended time periods and demands the use of a substantial list of meta-alerts.

#### *2.3.1.2 Building Scenarios from Heterogeneous Alert Stream*

Dain and Cunningham [9] use an alert clustering scheme that fuses alerts into scenarios using a "probabilistic in nature algorithm." In this system, scenarios are developed as they occur, i.e., whenever a new alert is received it is compared with current existing scenarios and then assigned to the scenario that yields the highest probability score. If the score falls below a threshold, it starts its own scenario. This testing is done in a time proportional to the number of candidate scenarios.

For alert comparison, the authors use a *Bigram model*, where the new alert is compared with the most recent alert in a scenario [9]. The probability of this is computed as a product based on three factors - all of which make use of alert categories discovery, scan, escalation, denial of service and stealth. The first factor is the *strength of link between two alerts*, which depends on the likelihood of the alert of a certain category following alerts of other categories. For example, the probability of an escalation type of alert followed by a DOS type of alert is less than the probability of a scan type of alert followed by an escalation type. The second factor is the *time between alerts* which is a sigmoid function based on the alert categories. For example, the time between DOS types of alerts might be small but can be more between discovery and escalation types of alerts.

The third factor is the *source IP address range* of the alerts, which also depends on the alert categories. The authors claim that this technique allows finding scenarios even if the attacker uses stealthy attack methods such as forged source IP addresses and long latencies between attacks. The authors also claim that combining only a few simple features is sufficient for satisfactory results.

For experimentation, the authors used *tcpdump* data from the DEFCON 8 hacker conference where participants take part in a game of attacking hosts in the network [9]. Although this data is rich in volume and content, it is an unusual set of data because it contains a large number of attacks coming from the same subnet in a small amount of time. The *tcpdump* data was replayed on a network using the Netpoke[4] tool and then the sensor RealSecure was used as an IDS to generate 16,250 alerts. A human expert categorized these alerts into 81 alert scenarios spanning a period of three hours. From the alert data, a data set containing both positive and negative examples was generated consisting of 793,167 training examples.

The authors used the alert data to estimate the parameters to be used in the probability estimation so that finding the likelihood of an alert in joining a scenario is optimized. A constrained optimization routine (fmincon, found in MATLAB) was used

[4] Netpoke: "Utility used to replay packets to a live network that were previously captured with the tcpdump program" http://www.ll.mit.edu/IST/ideval/tools/tools_index.html

for this purpose. After optimization by running probability estimation on data, it was found that, when compared with human experts' decision of joining, the tool produced 99.86% correct results for alerts that were not supposed to be joined in respective scenarios and 88.45% correct results for alerts that were supposed to be joined in respective scenarios.

#### *2.3.1.3 Alert Correlation with Hidden Colored Petri-Nets*

Yu and Frincke propose a model for Alert Correlation and Understanding (ACU) based on Hidden Colored Petri-Nets (HCPN) [63]. The model is based on the premise that earlier steps in intrusion prepares for the later ones.

In this work, the authors propose a modeling tool called Hidden Colored Petri-Nets as an extension to Colored Petri-Nets (CPNs). CPNs are used generally in modeling discrete event dynamic systems and have been applied in intrusion detection problems. HPCNs model agents, resources, actions, and functions of a system with transition and observation probabilities. Figure 2.3 shows a HPCN model for a L2R (LocalToRemote) attack where an attacker at a remote location gains local access to a machine [63].

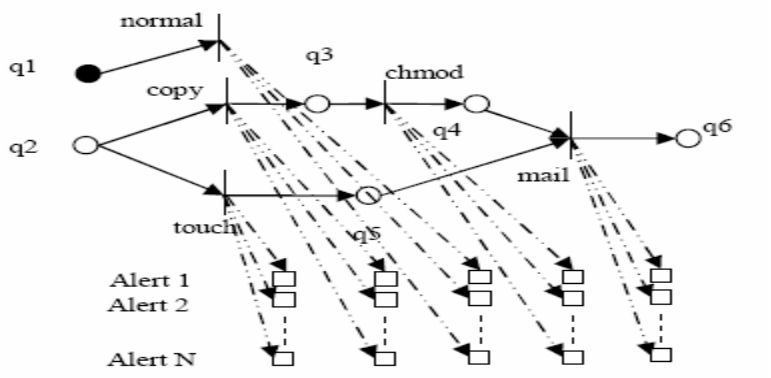


Figure 2.3 An Example HPCN Model (Taken from [63])

The transitions ( | ) in the chart denotes the actions in the attacks: normal, copy, chmod, touch, and mail. The nodes represent resources and the arcs represent prerequisites and consequences of each action. For example, with the q2 resource, the attacker can perform two actions that are copy and touch. After copy, the attacker can hold q3 and after touch the attacker can hold q4. An action can be observed from different alerts with different probabilities where the probabilities depend on false positive and false negative rates of the attacker's action. The model represents compromised resources instead of alerts [63]. Inference occurs through transition between actions as all prerequisite conditions are met for each action. As output, the model generates the probability of each resource being compromised as a confidence score.

To perform correlation, a model based on prerequisites and consequences was generated using domain knowledge. Also, probabilities of resources owned by agents are determined initially using policy and logon credentials. To better estimate the model parameters (i.e., observation and transition probabilities) needed to best explain known data, the authors use Moon's Expectation Maximum algorithm [63].

Yu and Frincke reported experiments on their HPCN-ACU model using the DARPA 2000 intrusion detection evaluation datasets and used RealSecure sensor's output to perform intra-sensor correlation [63]. The authors used part of the dataset as training set for estimation of the model parameters. The HPCN model for this data consisted of 20 resources, 20 actions and 28 alerts. The author evaluated their model by analyzing detection rates and false positive rates in terms of the number of compromised

resources identified. The authors reported a 100% detection rate and 20% false positive rate for the DMZ traffic and 92.31% detection rate and 0% false positive rate for the inside traffic. The authors further claim that ACU can improve the effectiveness of intrusion detection systems and be used to develop response plans. Although the authors state that this model can combine alert fusion (i.e., alert clustering) and intention recognition (i.e., alert correlation) at the same time in one system, alert clustering is not performed in its traditional sense, i.e. grouping together alerts with common features. The authors perform alert clustering in terms of identifying alerts that contribute to the same attack. This type of alert clustering is inherent with this model when different alerts can be held responsible for a single action.

#### *2.3.1.4 Alert Fusion based on Intrusion Reference Modeling*

The SCYLLARUS system, part of the ARGUS project developed at Honeywell Laboratories, integrates multiple intrusion detection systems (IDSs) into a unified architecture to provide overall intrusion situation awareness [15]. SCYLLARUS is primarily a knowledge-based expert system based on a description logic system with the focus on information modeling. The system is centered around an Intrusion Reference Model (IRM), which consists of the following [15]:

- *Network Entity Relationship Database (NERD):* contains configuration information of the site;
- *Security Goal Database*: contains security objectives and policies of the site; and
- *Event Dictionary*: contains information about both malicious and non-malicious events.

The authors assert that such an explicit model of the protected network enables sophisticated reasoning [15]. For such modeling, SCYLLARUS uses the CLASSIC object-oriented database system developed by Bell laboratories. The authors used CLASSIC for the following advantages - rapid prototyping, support of meta-data, support of multiple inheritance and automatic classification.

SCYLLARUS' embodies a *Dynamic Evidence Aggregator* that aggregates, correlates and fuses reports from multiple IDSs in a three-stage process [15]. At first, a *Cluster Preprocessor* reads the reports and generates hypotheses to explain them. Then an *Event Assessor* weighs the hypothesis according to their merits and identifies the plausible ones to record and dispose of hypotheses that are deemed to be implausible using expert judgment. The set of events and their evidence are used to construct a directed chart similar to a causal probabilistic network. The security goals that seem to be threatened by the plausible events are identified for impact assessment and finally an "understandable, goal-based summary" of the IDSs' reports is provided to the security administrator [15].

The Event Assessor plays a crucial part in the SCYLLARUS system. Although the authors state that the event assessor is based on qualitative probability theory, they did not explain how this technique is used to weigh plausible events. The illustration of example summary data does not show prioritization or comparative analysis of the sensor reports. As reported by Goldman et al. [15], the SCYLLARUS system is a proof-of-concept prototype that has not been thoroughly tested at the time of the report.

The system was tested on a particular multi-staged attack scenario with simulated IDS reports but the report did not describe the experiment or the results. The authors claim that this technique is useful for filtering false positives in multi-sensor systems.

#### *2.3.1.5 Probabilistic Intrusion Data Fusion*

Ye et al. [60] discuss fusion of sensor data as the process for determining a quantitative value for the system level such that the value is representative of the degree of concern in the system.

Ye et al. [60] denote data reported by intrusion detection systems as indication and warning (IW) values for the "component" under surveillance. Components are entities that are observed to identify intrusive actions. Examples of such components are: incoming packets to a particular host in the network or Telnet connections to a particular host in the network. To understand the interactive effects of network intrusions that involve co-coordinated actions, the IW values by themselves do not provide enough information since they only account for isolated effects of intrusion. The authors argue that it is essential to fuse *component level* IW values into *system level* IW values "to account for interactive effects of coordinated actions." The system level IW indicates the total effect of the components under observation. For example, host level IW will indicate the total effect of IWs for the monitored components of the hosts such as the total effects of the incoming packets to the particular host and the telnet connections to the particular host.

At first the IW values for a single component in the network, as generated by multiple different techniques, are fused into a composite IW value and then the composite IW values for all components in the network are fused [60]. To merge the IW values for a single observed entity in the network (for example, packets to a host in the system) into a composite IW value, a single object fusion technique based on probabilistic Dempster-Shafer theory (which does not require prior knowledge) is used. In this case, the IW values indicate the likelihood of the component being in an anomalous or compromised state as determined by the different sensors used for intrusion detection.

To merge the component level IWs into a system level IW, multiple value object fusion techniques based on Bayesian techniques are used [60]. In a Bayesian network for intrusion detection, the objects denote the components of the host under observation and the relations of objects denote their correlated states in intrusions. The authors obtain the structural relationships in the Bayesian network through training. The training data is obtained by simulation of normal and abnormal activities in the network. The authors use Chi-square tests to determine dependency of objects in order to generate the initial structure of the Bayesian network. Then, mathematical models are used to determine the prior probability tables of each object and the joint probability tables of each relation. After that, greedy and heuristic strategies are used to refine the Bayesian structures. Finally, composite IW values are calculated using the HUGIN evidence collection algorithm.

The authors illustrated use of their techniques using an example that monitors abnormality in the inbound packets to a host for a particular Telnet port and show how the component level IWs and the system level IW are computed. The authors do not describe how the training data was simulated and do not illustrate the nature of the Bayesian structure learned from the data. The nature of the relationships between the components and how these relationships relate to the system level IW value is not clear in this work.

### *2.3.2 Possibilistic Approach*

#### *2.3.2.1 Our Previous Work using Fuzzy Cognitive Modeling for Decision Support in Network Security*

A cognitive model is a "generalization over repeated experience" where knowledge acquired through perception and experience is organized into mental structures. One such cognitive modeling and inferencing technique uses *Fuzzy Cognitive Map (FCM)* that allows us to represent our perception of the real world in a structured way. FCMs, which originated from the synergism of fuzzy logic and neural networks, are an efficient soft computing tool. Soft computing has the human mind as the central focus and differs from hard computing by accommodating tolerance for impreciseness, uncertainty and partial truth [29]. A brief introduction to FCMs follows.

The idea for FCMs originated from cognitive maps, which were proposed by political scientist Robert Axelrod [2]. Later, Kosko introduced FCMs as signed directed graphs that model the world as concepts and causal relations between concepts in a structured collection [26, 27, 28]. *Concepts* (nodes) in an FCM (Figure 2.4) are events that originate

in the system and whose values change over time. The causality links between concepts are represented by directed *edges* that denote how much one concept impacts the other(s). The concepts in the FCMs can be crisp or fuzzy. Concepts typically take values in the interval [0,1]. In the simplest case, a concept is either on (1) or off (0). A concept can also be represented by a fuzzy set and can fire to some degree. The edges typically have values between 0 and 1 or –1 and 1. Edges can also be fuzzy, and in those cases we can use linguistic words such as "a little," "highly," "somewhat," to represent the edges. When the edges between concepts are fuzzy values, fuzzy set operators like T-norms and T-conorms can be applied to the particular chain of concepts to infer the total effects of concepts in the chain. Figure 2.4 shows two FCM concepts $C_i$ and $C_j$ connected by edge $e_{ij}$. The edge $e_{ij}$ can be used to define rules or causal flow between the concept nodes $C_i$ and $C_j$.

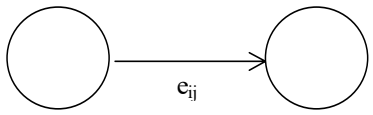


Figure 2.4 Two FCM Concepts and a Connecting Edge Representing a Causal Link

FCMs support adaptive behavior and provide a graphical representation of knowledge that can be used for explanation of reasoning [53]. Researchers have used FCMs for many tasks in several different domains. Among these are: fault management in a distributed network environment [41], disease diagnosis in the medical domain [54], and failure modes effects analysis [43]. Smith and Eloff reported their work on enhanced risk assessment in a health care institution using cognitive fuzzy modeling (a combination

of FCMs and fuzzy rule-based techniques) [51]. As described in [48], our use of FCMs was the first reported application in any area of network security including sensor alert fusion. It was a novel attempt to use FCMs for fusing alert information in a multi-sensor intrusion detection environment.

In the Intelligent Intrusion Detection System (IIDS)[5] architecture, we used Fuzzy Cognitive Modeling to provide the security administrator with an overall security view, where sensor fusion was carried out by fusing data reported by the sensors for a higher-level alert inference and by integrating inferences to assess the overall system health [4, 48, 49, 50]. Our decision making process underwent two levels alert inference, as shown in Figure 2.5 [50]:

1) At the initial level, Suspicious Events (SE) were identified for individual hosts and users across the network by combining reports from the sensors.

2) At the second level, the alert levels for each host and user (H/U) were computed by combining the results of the suspicious activities.

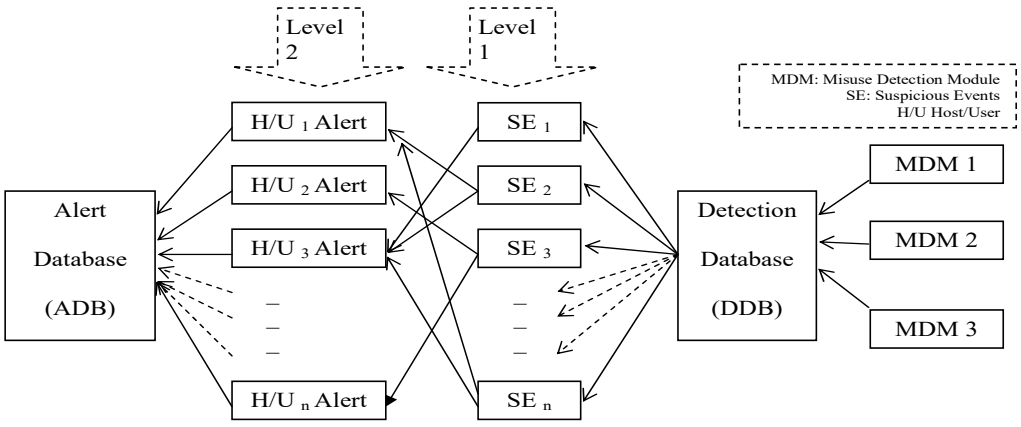


Figure 2.5 Individual Alert Generations for Hosts/Users

[5] A research prototype incorporating AI techniques for intrusion detection and developed at the Center for Computer Security Research (CCSR), of the Department of Computer Science and Engineering at Mississippi State University.

FCM models were used to identify the suspicious events by incorporating several FCM concepts for each kind of intrusion (for example, failed login attempts, finger bombs, SYN-Flood attacks). Suspicious events could be of different types that would affect the host and user alert level in specific ways. Each of these suspicious events could be triggered by analyzing the results of sensors and were activated with the help of a fuzzy rule-base where fuzzy rules were used to map multiple inputs to outputs. At the next level of sensor fusion, alerts for individual hosts and users were computed by combining evidence of suspicious activities affecting each of them. All activated suspicious events could impact the host and/or user alert levels in different ways. The degrees of impact varied depending on the nature of the suspicious event itself. Figure 2.6 shows an example of a subset of FCMs that we employed for evidence combination in the IIDS architecture [50].

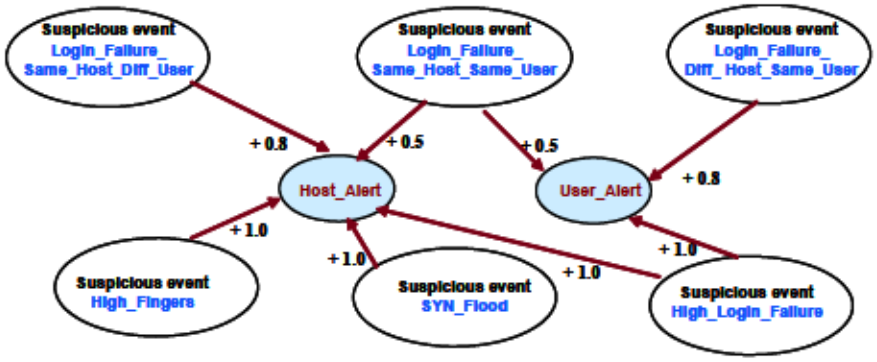


Figure 2.6 Combining Evidence of Multiple Suspicious Events

At the final stage, inference integration involved integrating the effects of the anomaly and misuse inferences to determine the overall network alert situation considering combinations of seemingly harmless events. The final deduction was

reported to the security administrator by sending and recording the results in the IIDS Alert Database. The security administrator monitored the network health through a Graphical User Interface (GUI) [14] that picked up and presented alert data in graphical form from the Alert Database. Controlled experiments [50] in a network environment demonstrated the feasibility of using a causal knowledge inference technique based on FCMs for sensor data fusion to aid in decision support.

#### *2.3.2.2 Fuzzy Intrusion Recognition Engine*

In 2003, the Fuzzy Intrusion Recognition Engine (FIRE), a network based intrusion detection system, reported use of FCMs in detecting attacks from features extracted from network traffic [59]. FIRE is a distributed IDS that employs independent agents for intrusion detection. Each agent is dedicated to monitor network traffic connections such as TCP, UDP, ICMP and various service ports. FIRE uses data visualization techniques to help in selecting key features to be used in the intrusion detection task. In the fuzzy rule-based expert systems employed by the agents, the features' characteristics are used to define fuzzy feature sets and expert rules. The rules are then used for misuse detection and any matched behavior is flagged as intrusion. FIRE employs monitors that combine reports from the transceivers or agents and correlate them to detect attacks.

In this research, Lincoln Lab's DARPA 1999 evaluation data was used [59] to support the experiments. Two weeks of data were collected for offline feature analysis. After analyzing the collected data with visualization, experts identified the features that

would be most effective to model for intrusion detection. The system was tested with fuzzy misuse expert rules and an FCM expert model as shown in Figure 2.7.

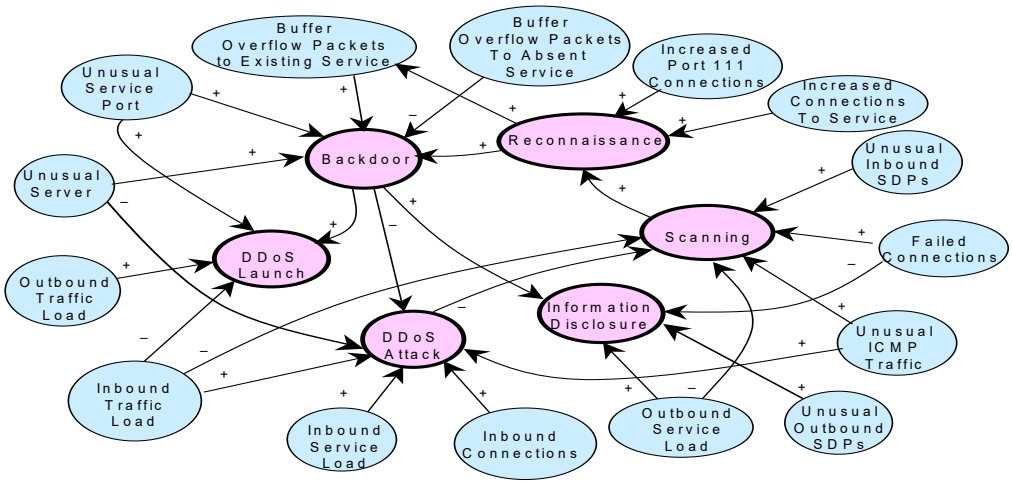


Figure 2.7 FCM Model to Detect Different Types of Attacks (Taken from [59])

The authors concluded that the FCM model correctly detected the different types of attacks.

# CHAPTER III

# RESEARCH APPROACH

## 3.1 Overview

This chapter describes the unified alert fusion model for intelligent fusion of sensor alerts in an intrusion detection environment. In this dissertation, we refer to *intelligent alert fusion* as the process of interpretation, combination and analysis of alerts reported by sensors to determine and provide a quantitative value for the system such that the value is representative of the degree of concern in the system. The possibilistic approach we employ uses *fuzzy cognitive modeling* for reasoning and *possibilistic information combination operators* for fusion.

## 3.2 The Unified Alert Fusion Model

The *unified alert fusion model* provides an overall condensed view of the network by assessing the health of the resources[6] in the network. The fusion model is *resource centric*, i.e., all analyses are centered upon the resources in the system. A resource centric view inherently reduces alert volume by:

- presenting an overall picture of the compromised resources to the security administrator instead of the large volume of all alerts issued;
- not having to take account of information that is out of the scope of the resource perimeter.

[6] A resource is essentially a host in a dedicated network of computers.

The input to the unified alert fusion model comes from the low-level IDSs that are treated as sensors. The sensor alerts are deposited in a central repository to be consumed by the unified alert fusion model. The alert reports are used in a way that does not conflict with the proposed standard Intrusion Detection Message Exchange Format (IDMEF) by the Intrusion Detection Working Group [20].

The unified alert fusion model incorporates the following functions (Figure 3.1):

- Prioritize alerts;
- Identify alert associations by:
  - clustering similar alerts; and
  - correlating related alerts;
- Assess the overall security situation.

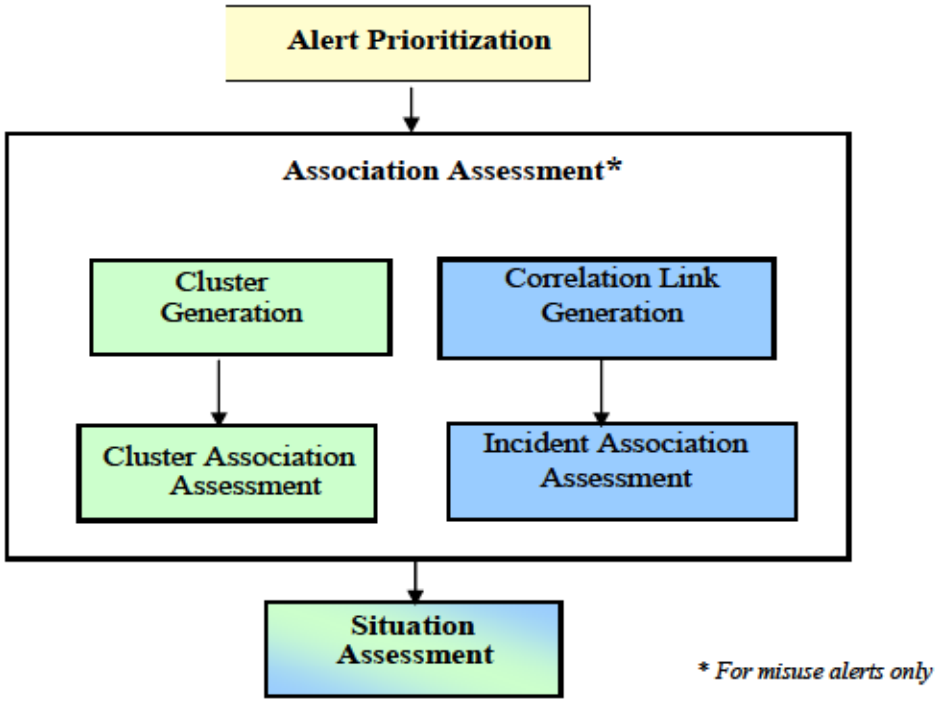


Figure 3.1 The Unified Alert Fusion Model

### *3.2.1 Alert Prioritization*

Alert prioritization is performed to assess the relative importance of alerts generated by the sensors[7]. The significance of an alert can depend on the following factors:

- *Source/Target criticality*: One of the most important pieces of information conveyed by an alert is that which denotes the resource targeted and the source the attack came from. Criticality can vary depending on the pairs of source/target in communication. For example, the highest priority is considered for communication when the source or the target lies inside the protected network perimeter and the other outside it. Again, being a resource centric model, the lowest priority is given for communication between pairs outside of the protected network. Therefore, significance of the source and the target resource considerably affects the assessment of the importance of the alerts.

- *Attack criticality*: Another extremely important piece of information conveyed by an alert is that which denotes the nature of the attack. Different attacks may have different levels of importance. For example, for a high performance cluster, availability attacks are considered more critical than confidentiality attacks. Therefore, the nature of the attack is taken into consideration for assessing the importance of alerts when it is available. This is particularly useful to consider for misuse sensors because a misuse sensor is able to detect the nature of the attack with pattern/signature matching.

- *Alert confidence*: The other important information that is taken into consideration while assessing the importance of alerts is the reliability of the alert itself. This is particularly useful to consider for anomaly sensors because an anomaly sensor typically associates a confidence value with the alert that is indicative of the sensor's confidence in declaring such an "out of the ordinary" pattern as an alert.

[7] As discussed in section 1.3 of Chapter I, sensors can be of two types: misuse (signature based) and anomaly (profile based).

Source/target and attack criticality measure the relative importance of the information itself in terms of the designated security policy[8], while alert confidence measures the trustworthiness in the information by the reporting sensor. For anomaly sensors, considering attack criticality does not make sense because anomaly sensors cannot detect the specific type or nature of the attack, other than reporting the manifestation of anomalies. On the other hand, for a misuse sensor, considering alert confidence does not make any difference because such sensors only report alerts with full confidence when they completely match attack patterns, i.e., there is no associated degree of uncertainty.

Based on the discussion above, for misuse sensors, the priority of an alert ($A_{pr}$) is computed as a product of the source/target criticality ($ST_{cr}$) and the attack criticality ($A_{cr}$), i.e.,

$$A_{pr}=\Pi(ST_{cr}.A_{cr});$$ and

for anomaly sensors, the priority of an alert is computed as a product of the target criticality[9] ($T_{cr}$) and the alert confidence ($A_{cnf}$), i.e.,

$$A_{pr}=\Pi(T_{cr}.A_{cnf}).$$

Once alert priorities are assessed, the fusion model filters out the lower priority alerts so that further analysis can focus on more important alerts. Often in real systems, predominant and persistent root causes behind systematic and repetitive false positives

[8] For the purposes of this work, the security policy for a system refers to the statement of information values and protection responsibilities for the system. E.g., the security policy for a certain system may designate that for that particular system, attacks that lend attackers access to system should be given more attention than those that attackers use for surveillance purposes.

[9] Since anomaly sensor reports do not contain source IP address information.

are caused by low priority alerts that can be attributed to a majority of all alerts generated in systems [22]. Therefore, prioritizing and filtering of alerts aid in substantial reduction of alert volume. This essentially makes the alert analysis process more attentive to prioritized alerts that are possibly malicious and less distracted by innocuous alerts. After alert prioritization, the alert fusion model proceeds with further analysis of alerts.

### *3.2.2 Association Assessment*

Debar and Wespi point out in [9] that alerts may not seem significant when they are isolated, but their importance may intensify when association(s) can be discovered among them. Therefore, after prioritization, the alert fusion model conducts *association assessment.* In general, associations can be of two types: structural (based on constituents of alerts) and causal (based on cause and effects of alerts) [37]. Structural association relates two (or more) alerts such that they have common alert contents, i.e., they convey similar information. Thus structural association is useful for identifying common patterns in attacks. The alert fusion model performs structural association with alert clustering. Causal association relates two (or more) alerts where one indicates an attack that sets the stage for the other(s) to follow. Thus causal association is useful for identifying alerts that can be attributed to multi-staged attacks. The alert fusion model performs causal association with alert correlation. In order to discover both structural and causal types of associations between alerts, the alert fusion model performs alert clustering and alert correlation on the same set of alerts independently, particularly for misuse sensor reports. This is because the information conveyed in alerts reported by anomaly sensors is not considered sufficient to find any structural or causal relationship between them.

Therefore in the case of anomaly sensors, the fusion model does not conduct alert association. Only the data reported by the anomaly sensors is directly used for final situation assessment. For misuse sensors, on the other hand, the results of causal and structural alert associations are combined for the overall situation assessment.

#### *3.2.2.1 Alert Clustering*

Alert clustering involves intelligently grouping or merging together identical alerts such that common generic attacks on systems are discovered [63]. In traditional clustering approaches, common feature values between two alerts are compared for a perfect match in order to consider them identical alerts. Alerts, whose features do not match exactly, are not aggregated. One of the objectives of this work is to extend alert cluster perimeters such that such alerts whose features are not exactly identical can be included in related clusters. This will provide the security administrator more insight into attack situations.

This dissertation proposes a *multi-level alert clustering* approach where alert features are clustered at different levels of abstraction or resolution such that different degrees of deviations in commonality are tolerated. In this way, along with identical alert clusters, clusters of "similar" alerts are also found. This dissertation applies a combination of an attribute-oriented generalization technique and a possibilistic approach with Fuzzy Cognitive Modeling to perform multi-level alert clustering. In this conceptual clustering approach, the cluster representations remain in the foreground and the model seeks clusters that match the representations [23].

The advantages of conceptual clustering are that intelligible descriptions of clusters facilitate cluster interpretation and conceptual clustering can handle categorical attributes like IP addresses and attack names.

For alert clustering, this dissertation uses the following set of characteristic features of sensor alerts:

- Source: This feature identifies where the alleged attack is coming from. If the source IP address is not spoofed, it can identify the attacker. This is a categorical attribute consisting of an IP address.
- Target: This feature identifies for whom the alleged attack is meant for, i.e., it denotes the target or, as referred to in this work, the resource to be protected. This is also a categorical attribute consisting of an IP address.
- Time: This feature denotes the time of the alleged attack. This is a categorical attribute denoting time stamps.
- Attack Name: This feature identifies the alleged attack. Different sensors often refer to the same attacks with different attack names. This is also a categorical attribute with textual attack names.

Clustering on source attributes can help to associate alerts originating from the same/similar sources. Clustering on time attributes can help to associate alerts that occur in short/close intervals. Clustering on attack names can help to associate alerts that are of the same/similar nature. Clustering involving categorical attributes is not straightforward because categorical attributes have domains that are discrete and unordered. In this respect, Julisch used taxonomies or generalization hierarchies of attributes to find clusters of alarms/alerts [22].

As mentioned before, our alert fusion model is *resource-centric* and therefore, alert clustering attempts to discover associations of each resource with identical alert clusters. The alert fusion model initially finds clusters of the same alerts by aggregating

alerts that correspond to an exact match of feature values. Then in order to find more clusters that are related, the fusion model trades specificity for generalization and generalizes the feature values to consider more alerts that are similar in some respect. The alert fusion model considers the notion of "similarity" in terms of category/class/type matching at different levels of generalization hierarchies. A *generalization hierarchy* (a.k.a. concept hierarchies) shows how concepts are organized into more general concepts [23]. The multi-level representation framework of generalization hierarchies organizes commonalities into tree structures [25]. The root in a generalization hierarchy provides general properties shared by all the descendents. The descendents have specialized properties that make them distinct among the siblings. In other words, the root of any subtree is a more abstract concept than any descendants of that subtree. Generalization hierarchies are useful tools to deal with categorical attributes [22]. For example, a categorical attribute, the attack name (i.e., name of the attack appearing in the alert) is one of the most crucial alert features. In a multi-sensor report, attack names can lead to confusion or redundancy because almost always, different sensors refer to the same attack differently. For example, while Snort IDS refers to the alert generated as a result of someone looking for the *sadmind* service in hosts as "*RPC_Portmap_Sadmind_Request_UDP*," RealSecure[10] IDS refers to the same attack as "*Sadmind_Ping*". Also, sensors sometimes refer to attacks that serve the same purpose with different attack names.

[10] RealSecure: A widely used commercial network intrusion detection tool developed by ISS, Inc. [21].

In order to gain more insight into attack names, a generalization hierarchy of attack names is used by the alert fusion model. Figure 3.2 shows the generalization hierarchy that was developed for this purpose[11].

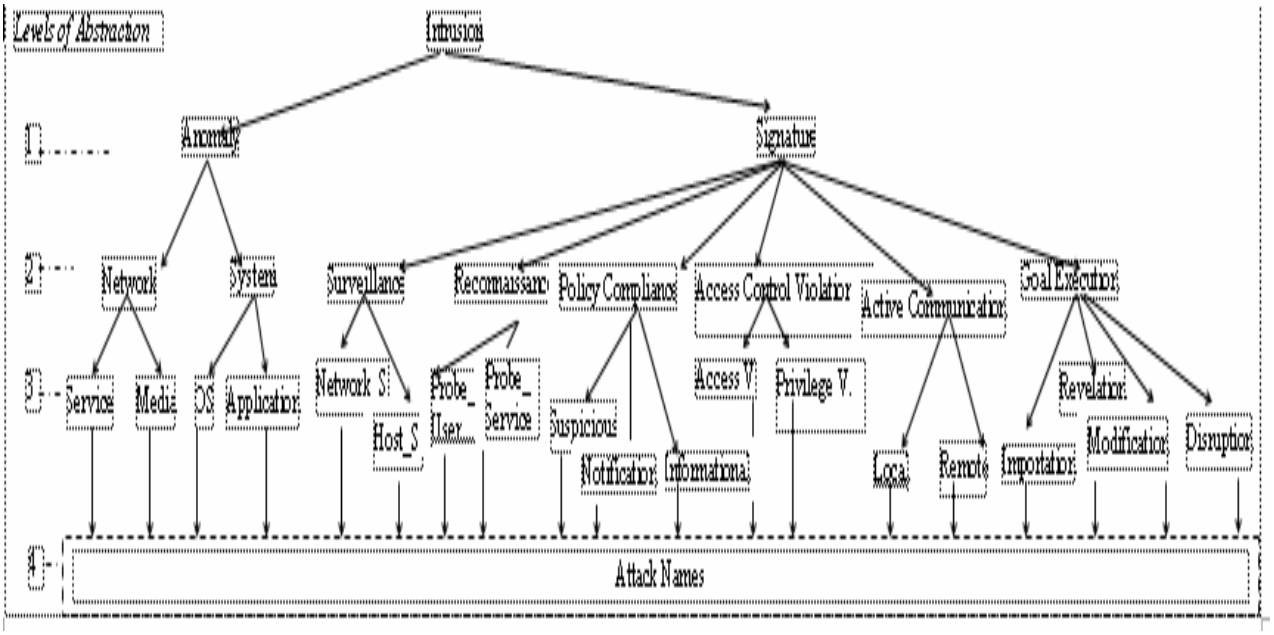


Figure 3.2 Generalization Hierarchy for Attack Names

Figure 3.2 shows how specific attack names are generalized into different abstract categories of attacks at different abstraction levels. In this generalization hierarchy, commonality of alerts is considered based on the nature of impact of the attacks that generate the alerts, i.e., it focuses on what the attacker achieves by executing the attacks on systems. The following is a description of the concept nodes[12] in the hierarchy:

[11] Some of the abstract concepts used in this hierarchy are adapted from [44].
[12] It should be noted that for the purpose of this work, the nodes in the generalization hierarchy are considered mutually exclusive. That is, an attack can belong to only one leaf node in the tree.

Surveillance: Designates alerts that are attributed to general activities which collect information about networks or systems. Surveillance activities are considered non-critical threats to the system but they may be used as preludes to conducting specific attacks on systems. Surveillance can be further refined as:

- *Network_Surveillance*: Designates alerts for surveillance activities on a network that specifically attempt to discover the existence of active systems in a network (for example, alerts for *IPSweep* and *Subnet Sweep* attacks).

- *Host_Surveillance*: Designates alerts for surveillance activities on a particular system that specifically attempts to confirm the existence/active status of the system (for example, alerts for *Ping*).

Reconnaissance: Designates alerts that can be attributed to activities that collect specific information about networks or systems. Reconnaissance activities are considered non-critical threats to the system but they may be used to conduct further malicious acts which may cause harm to systems. Reconnaissance can be further refined as:

- *Probe_of_Service*: Designates alerts for reconnaissance activities that are targeted to a particular system to specifically obtain information about specific services supported by the system (for example, alerts for *Port Scan*, *Ping of Service*).

- *Probe_of_User*: Designates alerts for reconnaissance activities that are targeted to a particular user to specifically obtain information about user accounts (for example, alerts for *Finger*).

Policy Compliance: Designates alerts that are attributed to general non-intrusive activities that are reported in compliance with the security policy of the system. Policy compliance alerts indicate activities that do not pose any immediate threat to the system or cause any immediate impact on the system. Policy compliance is refined into:

- *Notification*: Designates alerts, which are reported for events that are significant according to security policy (for example, alerts indicating that a vulnerable service is running in a system, or alerts indicating attempts to initiate communication sessions between systems).

- *Suspicious*: Designates alerts, which are reported for atypical events that are considered suspicious (for example, alerts indicating failed access to systems, alerts indicating undesired use of system protocols).

- *Informational*: Designates alerts that indicate release of non-critical information related to systems (for example, alerts indicating disclosure of directory or configuration information).

Access Control Violation: Designates alerts attributed to intrusive activities that compromise the system security perimeter. Examples are exploitation or manipulation of weak/insecure/inadequate system features or configuration/implementation errors to gain access to a system. Access control violation activities pose an immediate critical threat to the system and may cause further impacts on the system. Access control violations are further refined into:

- *Access_Violation*: Designates alerts that indicate specific intrusive activities that have the potential to lead to local user level access to systems with the ability to execute normal user commands (for example, alerts for *Dictionary* and *Guest* attacks).

- *Privilege_Violation*: Designates alerts that indicate specific intrusive activities that have the potential to lead to root level access to systems or privilege escalation from user level to root level - imparting total control of system (for example, alerts for *Fdformat* and *Phf* attacks).

Active Communication: Designates alerts that are attributed to general suspicious activities which open a communication channel between systems that may be used to transfer files to and from those systems. Active communication activities are considered to pose an immediate critical threat to the system and may be used for further attacks. Active communication can be further refined as:

- *Local*: Indication of a suspicious local communication channel (for example, alerts for *command shell*).

- *Remote*: Indication of a suspicious remote communication channel (for example, alerts for *remote shell* and *ftp transfer*).

Goal Execution: Designates alerts that indicate malicious attacks that unconditionally conflict with the security policy and have the potential to cause a system to behave in an unwarranted way. These activities are considered the most critical threats to a system. Goal execution is refined into:

- *Importation*: Designates alerts that specifically point to prohibited/illegal tool installation or usage in the protected environment to launch further attacks against or from the system (for example, alerts for *Mstream_Zombie* or *Trin00 DDoS* attacks).

- *Revelation*: Designates alerts that specifically indicate attacks used to compromise the confidentiality of any protected system asset such that the asset is exposed or disclosed to an unauthorized party.

- *Modification*: Designates alerts that specifically indicate attacks used to compromise the integrity of any protected system asset such that the asset is tampered with by an unauthorized party (for example, Tripwire *integrity* alerts).

- *Disruption*: Designates alerts that specifically indicate attacks used to compromise availability of a protected entity such that the system asset becomes unusable. Disruption attacks can make a system unusable by crashing it (for example, *Teardrop* attack), slowing it down (for example, *Smurf* attack), by exhausting its resources (for example, *Syn Flood* attack), executing unauthorized activity on or using the system (for example, *Virus/Worms*, *DDoS* attacks).

The use of such a generalization hierarchy allows the fusion model to find similarity between alerts with different attack names by generalizing them into abstract categories. For example, at level 4 (the most specific level) of the generalization hierarchy in Figure 3.2, two attacks *ffbconfig* and *fdformat*[13] may seem different, but at

[13] Attacks associated with root-owned *ffbconfig* and *fdformat* utility programs to gain root privilege.

level 3 they can be considered similar as they both belong to the *Privilege_Violation* category. Again, the fusion model can find similarity between the *dictionary*[14] and *ffbconfig* attacks at level 2, as they both belong to the *Access_Control_Violation* category. The fusion model considers alerts more similar that are found to match at level 3 than those that match at level 2. To capture this notion of similarity being directly associated with the level of the generalization hierarchy, the fusion model uses the distance between the abstraction levels to compute the feature similarity. In this respect, the fusion model makes use of *domain generalization paths* related to the generalization hierarchy for the alert features. Domain generalization paths can map the levels of a generalization hierarchy to a more general representation of the concepts at the same level [17]. The nodes at each level of the domain generalization path present a more abstract description of the concepts than the nodes at the lower levels. For example, the domain generalization path for attack name in Figure 3.3 represents the generalization hierarchy of Figure 3.2.

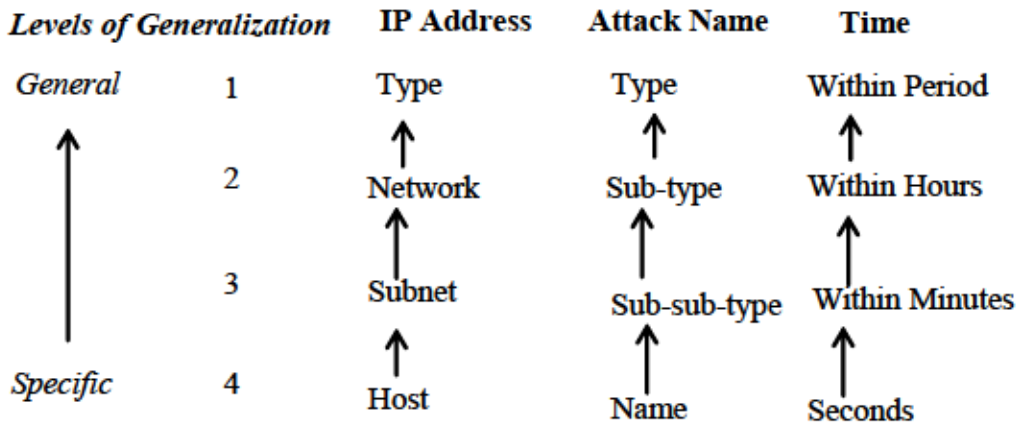


Figure 3.3 Domain Generalization Paths for Alert Features with Similarity Score

[14] Attack to crack password to gain user access.

For feature similarity determination, the alert fusion model uses the generalization levels of Figure 3.3. The most specific concept level (level 4), indicates the highest score (i.e., 4), applicable in this case. The score decreases by one as one moves from a specific to a generalized level. The most general concept level has the least score, which in this case is, 1. To illustrate this feature similarity determination, let us suppose two alerts have the same IP address. This will yield an exact match between the source features and a similarity score of 4 will be assigned for the source feature. Now, let us suppose the two alerts do not have an exact match of IP addresses. In that case, the fusion model will generalize the alert features at different levels to see if any match can be found at the higher-levels of abstraction. If found, the fusion model will score the alert features such that higher scores are given to matches at more specific levels and lower scores to matches at more generalized levels. Therefore, if the two IP addresses were found to be generated from the same subnet, source feature similarity would be given a score of 3. If they matched at only type level (for example, same class A type address), the score given would be 1. Other candidate features (like, attack names and time) are also matched and scored similarly.

With multi-level alert clustering, the fusion model seeks different types of suspicious identical clusters according to different combinations of the alert features and their abstractions. Table 3.1 shows the different clusters with their respective generalization information and the collective feature similarity scores that an alert contributes to belong in the respective cluster.

Table 3.1 Features of Suspicious Clusters

| Suspicious_Clusters | Source | Attack | Time | Max Score | Min Score |
|---|---|---|---|---|---|
| Same_Source_Same_Attack_Same_Time, $C_4$ | S | S | S | 12 | N/A |
| | | | | | |
| Similar_Source_Same_Attack_Same_Time, $C_{31}$ | G | S | S | 11 | 9 |
| Same_Source_Similar_Attack_Same_Time, $C_{32}$ | S | G | S | 11 | 9 |
| Same_Source_Same_Attack_Recent_Time, $C_{33}$ | S | S | G | 11 | 9 |
| | | | | | |
| Similar_Source_Similar_Attack_Same_Time, $C_{21}$ | G | G | S | 10 | 6 |
| Same_Source_Similar_Attack_Recent_Time, $C_{22}$ | S | G | G | 10 | 6 |
| Similar_Source_Same_Attack_Recent_Time, $C_{23}$ | G | S | G | 10 | 6 |
| | | | | | |
| Similar_Source_Similar_Attack_Recent_Time, $C_1$ | G | G | G | 9 | 3 |

The first cluster in Table 3.1 employs traditional exact clustering and does not allow deviations or generalization of any alert features. All the other clusters employ multi-level alert clustering with inexact matches of feature attributes. The second set of clusters in Table 3.1 involve generalization of only one of the alert features, the third set of clusters involve generalization of any two of the alert features, and the last one involves generalization of all alert features under concern. In the columns of Table 3.1 titled *source, attack* and *time*, S denotes specific or exact matching, G denotes matching with generalization.

When an alert generated for a particular resource (target of the attack) becomes a candidate for a cluster, the score it contributes to the particular cluster, is computed by taking into account the collective feature similarities of the particular alert with other alerts in the cluster. This *candidacy score* of an alert for a particular cluster denotes the extent to which the alert belongs to that particular cluster. In Table 3.1, the *Max* column represents the maximum collective score that an alert can contribute in this respect and

the *Min* column represents the minimum. To illustrate, consider the cluster *SimilarSource_SameAttacks_SameTime*. For this cluster to exist, the *attack* and the *time* features need to be exact and the only feature that can be generalized is the *source* feature. The score that results from alerts in this cluster will always be (4+4=8) for the *attack* and *time* features and vary in between 1 to 3 for the *source* feature (depending on at which level of generalization match was found). Therefore, for this specific cluster, alerts can contribute scores in the range of 9 to 11 (Min value, 8+1=9 and Max value, 8+3=11). Since an alert's candidacy score for a cluster can vary on a scale between 3 to 12 for all clusters (Table 3.1), the fusion model fuzzifies the crisp score by mapping it to a fuzzy variable with a normalized range of 0 to 1. In this respect, the crisp scores are fuzzified and aggregated according to the most widely used fuzzy models in practice, the Mamdani model[15] [62]. Figure 3.4 shows the complete term set of the fuzzy variable candidacy score, superimposed on the cluster score distributions for all possible clusters. To derive a crisp output for the candidacy score, a centroid defuzzification method [62] is employed.

[15] Developed by E.H. Mamdani, a widely used fuzzy inference model that use superimposition to derive conclusion of multiple rules into a final conclusion.

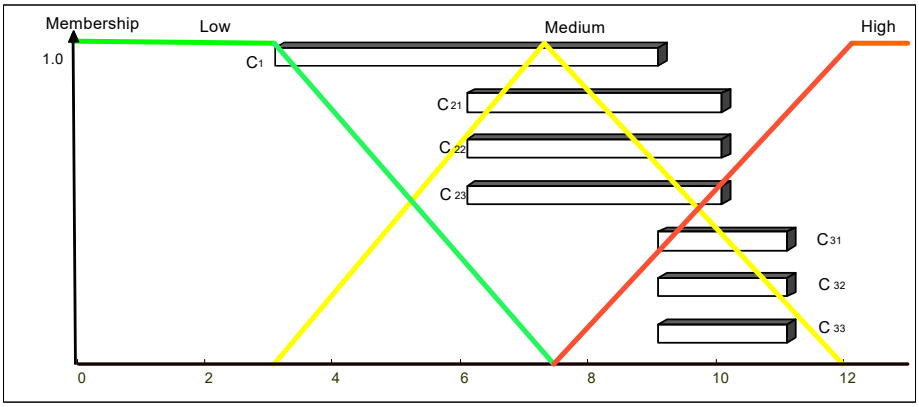


Figure 3.4 A Complete Term Set for the Fuzzy Variable Candidacy Score Superimposed on the Cluster Score Distribution

It should be made clear that the fusion model considers an alert to be a member of only one cluster. While the cluster definition allows one alert to be a member of several clusters, for simplification purposes, the fusion model does not allow overlapping and places a particular alert in the most specific cluster it relates to.

After the alert fusion model determines alert clusters for each resource in the network by aggregating alerts with the same and similar features, the strengths of the clusters are computed. *Strength* of a particular cluster is indicative of the "closeness" between the alerts in the cluster in terms of similarity of features. In order to compute the cluster strengths and also to fuse the overall impact of the different clusters activated for each resource, this dissertation uses cognitive modeling with Fuzzy Cognitive Maps (FCMs). FCMs were described in section 2.3.2.1 of Chapter II. For inference, the resemblance between FCMs and neural networks is utilized [5]. In the neural network approach, the concepts of the FCMs are represented by neurons and the edges are represented by the weights of the connecting neurons. The concepts, treated as neurons, trigger activation of

alert levels with different weights. Here, the concepts denote the clusters generated and the weights denote the contributing alerts in a particular cluster. Therefore, cluster strength is computed as an average of the candidacy scores of all contributing alerts in a cluster at each level of similarity. That is, for each level of similarity $l$ in a cluster $C_j$, if $N_l$ is the number of alerts clustered in that level, and $C_l$ is the candidacy score of the alerts in that level, then the strength of the cluster $C_j$ at time $t_{n+1}$ is:

$$S(C_j)(t_{n+1}) = \left[ \frac{\sum_{l=1}^{n} (C_l)(t_n) * N_l(t_n)}{\sum_{l=1}^{n} N_l(t_n)} \right]$$

Finally, a quantitative evaluation for the association of the resource with such suspicious clusters is computed according to the FCM model shown in Figure 3.5.

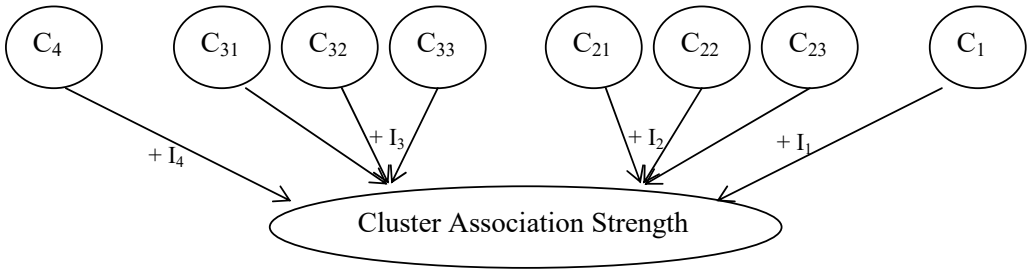


Figure 3.5 FCM Model for Combining Evidence of Suspicious Clusters

Figure 3.5 shows how different generated clusters affect the overall cluster association alert level of a resource. The impact levels are different depending on the type of the clusters. More specific type clusters cause more impact on the resource's association than less specific ones - according to the abstraction level used for generalization. Clusters involving more generalization would have less impact on cluster

association and vice versa. Therefore, it is always true that $I_4 \geq I_3 \geq I_2 \geq I_1$, where I indicate the extent of impact at the four different levels of generalization. The impact values are determined at execution time by using the maximum fuzzy candidacy score for each cluster. At any time, the cluster association alert level or the *Cluster Association Strength (CAS)* for a particular resource collectively represents its association with all the suspicious clusters activated for that resource at that time. In an FCM, like the one shown in Figure 3.5, the runtime operation is observed by determining the value of the effect concept from the cause concepts and the connecting edge values. Therefore, in accordance with FCM inference [28], CAS of a resource $R_i$ at time $t_{n+1}$ for each associated cluster $C_j$ of strength $S(C_j)$ and with impact $I_{ji}$, is represented by the following:

$$CAS(R_i)(t_{n+1}) = \left[ \frac{\sum_{j=1}^{n} S(C_j)(t_n) * I_{ji}(t_n)}{\sum_{j=1}^{n} I_{ji}(t_n)} \right]$$

The high-level steps for the multi-level alert clustering approach are shown below:

```
for each host x in X (X: host list for the protected environment)
{
   get all alerts into A that involve any communication with x
   find clusters with the same alert attributes
   generalize specific alert features to abstract alert features
   find clusters with similar alert attributes at different levels
   of generalization
}
for each host h in H (H: hosts reported with clustered alerts)
{
   for each cluster
       compute Cluster Strength
   compute total Cluster Association Strength
}
```

With multi-level alert clustering, the alert fusion model clusters alerts with the same and similar features. Cluster Association Strengths (CAS)s are also reported along with alert clusters for each resource. It should be pointed out that CAS represents a confidence score provided by the fusion model as an indication of the degree of concern for a particular resource's involvement in common attack patterns.

#### *3.2.2.2 Alert Correlation*

Alert correlation involves discovering causal relationships between alerts such that alerts that are associated in multi-staged attacks can be linked together. With the premise that every cause is bound to have an effect – whether the effect is critical or non-critical, the alert fusion model views the alerts generated by the sensors as causes with the potential to generate various impacts or effects in systems. Different alerts in sensor reports relate to different actions of the attackers which may have different objectives. The effects generated can potentially be coupled together in a causal chain to reveal the possible correlations between the alerts that initiate them.

This dissertation uses cognitive modeling with FCMs to represent different events in the system and the nature of relationships between them. To illustrate a common attack such as distributed denial of service (DDoS) is examined. Suppose, a DDoS attack is to be launched using a known vulnerability of the *sadmind* service in Solaris systems. In this case, the following steps are usually carried out by the intruder [35]:

- Execute *IPSweep* from a remote site to find the existence of hosts;
- Probe the hosts looking for *sadmind* daemon running on Solaris hosts;
- Break into host(s) using the *sadmind* vulnerability;

- Install *Trojan mstream* DDoS software; and
- Launch the DDoS.

Figure 3.6 is an FCM that models the DDoS attack scenario described above using cause and effect types of events (the events shown in this example are similar to the consequences of hyper alert types in [37]). The FCM model denotes that when the fusion model finds an *IPSweep* alert in the sensor report, it generates an *IPSweep* event, from which the *HostExits* event can be inferred. Later, when the *SadmindPing* event is generated from the alert report, the fusion model can associate this with previously generated *HostExits* event and then, both events together contribute to generate a new event *VulnerableToSadmind*. All alerts contributing to events of a particular FCM model can be correlated as part of the attack scenario depicted by the FCM model.

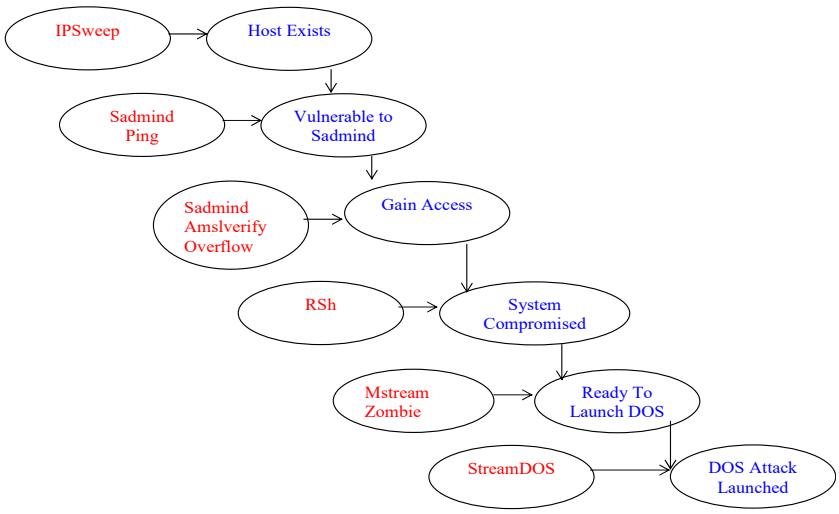


Figure 3.6 FCM Model for Detecting DDoS Attack with Sadmind Service Vulnerability

Figure 3.6 uses specific concepts to model the scenario described earlier. However, problems with such specific/exact knowledge modeling are that:

- it does not scale well; and
- it does not work well when there are deviations in data (due to heterogeneous sources) or incomplete data (due to incomplete/imperfect source coverage).

Therefore, in order to address these issues and make FCM models more applicable to real world situations, the fusion model employs abstract or more generalized cognitive models such that:

- a particular model can accommodate variations of similar knowledge or evidence; and
- inference can still take place with incomplete information.

In this regard, we employ *abstract incident modeling* where generalized events are used for alert correlation. Such abstract incident models can be developed to capture the essence of typical or commonly occurring techniques used by the attackers in multi-staged attacks focusing on the effects of the intrusions. In this research, we consider only direct effects of intrusions. For example, when an alert indicates evidence of an attacker exploiting certain weaknesses in a system that has the potential to grant local user access to the system, then only the local user access is considered as the after effect of the intrusion. It is possible that a local access can lead to further access in the system and that once a system's security perimeter is breached, greater damage can be afflicted; however, being conservative, the incident model simply considers the direct immediate effects and refrains from further reasoning until evidence of more related activities surface.

An abstract incident model can capture typical generalized attack patterns such as those shown in Figure 3.7 that show how different attacks in a system facilitate other attacks - all being part of coordinated multi-staged attacks. In an abstract incident model, an event can cause other events to occur or it can occur because of other events occurring in the system. Primarily, there can be two types of events activated in the system: *Cause Events (CEvents)* and *Effect Events (EEvents)*. The difference between the two types of events is that, as the names reflect, *CEvents* essentially contribute to *EEvents* or *EEvents* are activated by *CEvents*. The events in the abstract incident model of Figure 3.7 are described below:

- The events at far most left of the model (colored blue) are considered *CEvents,* which are generated as a result of alerts seen in the sensor reports and correspond to possible actions taken by the intruder to achieve some goal.
- The events at middle of the model (colored yellow) are considered:
    - *EEvents,* when they are generated as combined effects of the *CEvents* that correspond to the sensor alerts and the *CEvents* that correspond to existing risks in systems. *EEvents* are security incidents indicating a possible security violation in the system.
    - *CEvents* when they contribute to generation of risks of security incidents.
- The events at far right of the model (colored green) are considered:
    - *EEvents,* when they are generated as effect of the *CEvents* that correspond to some security incidents that have occurred in the system. Although not shown here, other external factors like, vulnerabilities or threats can also contribute to the activation of these events.
    - *CEvents,* when as risks they contribute to generate other security incidents in the system.

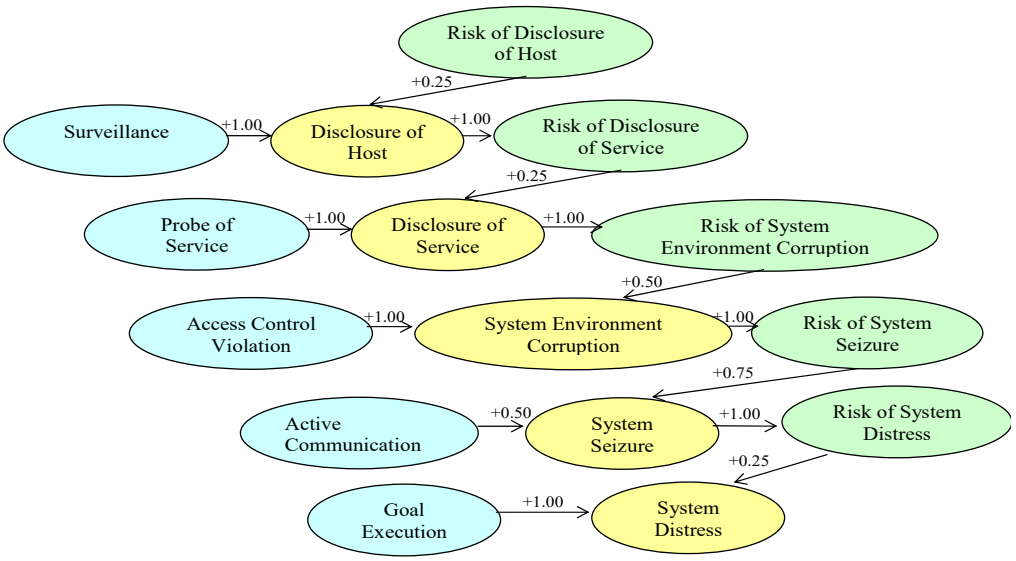


Figure 3.7 An Abstract FCM Incident Model for Multi-Staged Attacks in General

The leftmost *CEvents* in the abstract incident model of Figure 3.7 correspond to the generalized alert types in the attack name generalization hierarchy of Figure 3.2, a description of which is given in section 3.2.2.1. The middle *EEvents* in the abstract incident model of Figure 3.7 correspond to various security incidents and are described below:

- Disclosure_of_Host (DHS): This event occurs when there is evidence that a resource's identity is exposed or disclosed to outside users. *Surveillance CEvent* triggers this event in system. The knowledge about the existence of the resource can be used by the intruder in further probing to gain additional information to continue with additional attacks.

- Disclosure_of_Service (DSV): This event occurs when there is evidence that existence of a particular service resided on a particular resource is revealed and when there is pre-existing risk of such disclosure present. *Probe_of_Service CEvent* triggers this event in the system. The knowledge of the existence of a particular service that has known vulnerabilities/weaknesses can be exploited by the intruder for further attacks.

- System_Environment_Corruption (SEC): This event occurs when there is evidence of activity that may result in unauthorized access to resources such that the resource's security perimeter is breached and when there is pre-existing risk of such activity present. *Access_Control_Violation CEvent* triggers this event in system. Once the security perimeter is breached, an intruder can take necessary measures to attack the resource itself or to use the compromised resource to launch further attacks against other resources in the network.

- System_Seizure (SSZ): This event occurs when there is evidence of an unauthorized communication channel with the resource in question indicating total control over the resource and when there is pre-existing risk of such activity present. *Active_Communication CEvent* triggers this event in system. With transferring necessary files or tools, the intruder can proceed to attack the resource itself or use the compromised resource to launch attacks against other resources in the network.

- System_Distress (SDT): This event occurs when there is evidence of definitive malicious attack and when there is pre-existing risk of such an activity present. *Goal_Execution CEvent* triggers this event in the system.

All these events are linked together by cause and effect relationships in the abstract incident model of Figure 3.7. For a DHS incident, there is no predecessor incident in the correlation chain and all other DSV, SEC, SSZ and SDT incidents are considered its successors. For a DSV incident, the predecessor incident in the correlation chain is DHS and SEC, SSZ and SDT incidents are considered its successors. For an SEC incident, DHS and DSV incidents are considered its predecessors and SSZ and SDT incidents are considered its successors in the correlation chain. For an SSZ incident, DHS, DSV and SEC incidents are considered its predecessors and an SDT incident is considered its successor. For an SDT incident, all other DHS, DSV, SEC, and SSZ incidents are considered its predecessors and it has no successor incident in the correlation chain.

The following properties hold for inference with the abstract incident model:

- Suppose, $I_P$ denotes a predecessor incident and $I_S$ denotes a successor incident in a correlation scenario found for a resource $R_i$. If the earliest occurrence time of alerts contributing to $I_P$ is $t_{start\text{-}ip}$, latest occurrence time of alerts contributing to $I_S$ is $t_{end\text{-}is}$, occurrence time of an alert contributing to $I_P$ is $t_{ip}$, and occurrence time of an alert contributing to $I_S$ is $t_{is}$, then the following must be true for the alerts to be correlated:
  - *with predecessor incident*
    - $t_{is} => t_{start\text{-}ip}$
  - *with successor incident*
    - $t_{ip} <= t_{end\text{-}is}$
- For $I_P$ and $I_S$ to be correlated, they must occur between the same pair of hosts with one of them being the resource in question. (An exception is a *System_Distress (SDT)* incident, which may involve the resource in question with any other host. This is because once a multi-staged attack proceeds to the *SDT* level, alerts can designate definitive attacks directed to the host in question (e.g., *DDOS_shaft_handler_to_agent*[16] alert) or alerts can designate attacks originating from the host in question (e.g., *Mstream_Zombie_Response*[17] alert). Nevertheless, in both cases, the host in question is considered as the target of the attack.)

For alert correlation, evidence in the sensor reports (i.e., sensor generated alerts) are initially generalized to abstract alert types (as shown in the alert generalization hierarchy of Figure 3.2) and then mapped to the leftmost *CEvents* as shown in Figure 3.7. For example, if there is an alert that indicates a *sadmind* buffer overflow, then instead of generating a specific event like *SadmindAmslverifyOverflow*, as in Figure 3.6, the fusion model uses abstract alerts to activate more generalized *CEvents* like *Access_Control_Violation,* as in Figure 3.7.

[16] This is an alert generated by Snort denoting that a DDoS Shaft handler is directing a DDoS Shaft agent (compromised host) to launch an attack (http://www.snort.org).

Note that the same *CEvent* will also be generated for similar types of alerts such as *StatdOverflow*[18] or *SolarisLPDOverflow*[19]. Instead of generating specific events like *VulnerableToSadmind*, as in Figure 3.6, the fusion model activates more generalized *EEvents* like *Disclosure_of_Service,* as in Figure 3.7. Note that this same *EEvent* can also replace other specific events like *VulnerableToStatd,* or *VulnerableToSolarisLPD.*

Different actions of an attacker, targeted at a particular resource, activate different incidents for that resource. The extent to which such an incident occurs depends not only on the evidence of the corresponding action taken by the attacker as found in the sensor generated report, but also on the existing risk of such an incident taking place (Figure 3.7), i.e., the determination of what has happened jointly depends on what was reported to have happened (i.e., current evidence of the incident) and what could have happened (i.e., the possibility of the incident). For example, the *EEvent, System_Environment_Corruption* primarily depends on the sensor reporting of the *CEvent, Access_Control_Violation* (alert impact[20] designated by an FCM edge of +1.00). This type of action is not always successful and therefore, sensor notification of this alert does not guarantee that such an incident actually took place. The abstract incident model

---

[17] This is an alert generated by RealSecure denoting that an mstream agent/zombie (compromised host) is responding to an mstream handler/master (http://xforce.iss.net/xforce/search.php)

[18] *StatdOverflow*: An attack that exploits vulnerability associated with Solaris system's *statd* program that provides network status monitoring and crash and recovery functions.

[19] *SolarisLPDOverflow*: An attack that exploits vulnerability associated with *Solaris BSD print protocol* daemon.

[20] The edge values used in the correlation model come from security experts' common sense judgment and experience. Note that edges represent how much a certain concept impact the other, on a scale of 0 to 1 or 0 to -1. Although these impact values are determined from expert knowledge and experience, once the values are initially set, their performance can be observed over time and their values can be tuned for optimal performance by the security administrator based on the empirical performance of the alerts generated. We have found that FCMs offer a highly flexible structure in this regard. A variety of both manual and automated techniques can potentially be used to fine-tune these parameters.

deals with this uncertainty by taking additional information into account, i.e., the pre-existing risk of such an incident happening for the particular resource in question as shown by the middle *CEvents* in Figure 3.7. It shows this risk impact designated by the FCM edge of +0.50 for the incident *System_Environment_Corruption*. Note the difference between alert and risk impact. This is because security administrators tend to pay more attention to the report of the alert itself than to its existing risk. Sometimes when alerts such as - *rsh*, *Telnet XDisplay, Ftp_Put* (which generate *Active_Communication CEvent*) are issued by sensors, the existing risk (or possibility) of such an incident occurring impacts the incident more than the alerts do since such alerts are not always indicative of actual malicious activities. Hence impacts of such *CEvents* are less than the impacts of the associated risks. It should be noted that such risk computation can also incorporate other characteristic features of the resource itself such as the presence of known vulnerabilities in the host that can be exploited to cause security incidents. For example, if a resource is known to have the *sadmind* service running, thus making it vulnerable to a buffer overflow type of attack, this would increase the risk of the incident *System_Environment_Corruption* for that resource.

All alerts contributing to *CEvents* of the abstract incident model can be correlated as part of a general multi-staged attack scenario denoted by the FCM model. The following are the steps for such alert correlation:

```
for each host x in X (X: host list for the protected environment)
{
   get all alerts into A that involve any communication with x
   for each alert a in A
   {
      generate CEvent for the alert type
   }
   get list of each host y in Y that are in communication with x
   for all alerts that involve communication between x and y
   {
      generate EEvents (incidents and risks) and correlate alerts
          check for isolated alerts
          check for isolated non-critical incidents
      identify x as compromised
   }
}
for each host h in H (H: hosts reported with correlated alerts)
{
   for each incident
         compute Incident Strength
   compute total Incident Association Strength
}
```

It should be noted that a correlated scenario is unique between the pair of hosts involved in the communication. Multiple scenarios can be activated for one victim host because it is feasible for a host to become the target of a coordinated attack launched from multiple attack sources. In that case, the coordinated scenarios are reported differently depending on the source of the attacks and the nature of the attacks. When correlating multiple alerts for multi-staged attacks scenarios, isolated incidents (i.e., if alert correlation results in only one type of incident) are disregarded intuitively. An exception is the *System_Distress* incident, which may be generated due to isolated attacks such as *smurf* or *syn_flood*, and therefore even if isolated, it is reported because of its critical nature.

With FCM modeling of system events, presence of all predecessor events in the abstract incident model is not necessary to infer all subsequent events. For example, if a sensor does not report an *Access_Control_Violation CEvent* (which is possible because

"false negatives" are a common problem for sensors), a *System_Environment_Corruption EEvent* is still activated to some extent. This is because once *Disclosure_of_Service EEvent* gets activated in the system as a result of the *Probe_of_Service CEvent*, it activates *Risk_of_System_Environment_Corruption* immediately. The *Risk_of_System_Environment_Corruption EEvent* consequently activates the *System_Environment_Corruption EEvent* to some extent (not in full because some of the evidence of the incident is missing). This eventually causes other subsequent *EEvents* to activate partially as further inference takes place. As the situation builds and more associated alerts are reported, resulting *EEvents* become stronger. Therefore, alert correlation is able to progress to a partial extent with missing alerts in the sensor reports. Thus, use of abstract incident modeling allows the fusion model to replace multiple explicit attack models and helps with scalability and uncertainty issues in alert correlation.

Along with correlating alerts, the fusion model also reports security incidents that occur for the resources. The extent to which a particular security incident occurs designates its *incident strength*. In accordance with FCM inference [28], the strength of a successor incident $I_s$ activated for a resource $R_i$ at $t_{n+1}$ time for each predecessor incidents $I_p$ with impact $e_{pi}$, can be represented by the following:

$$S(I_s)(t_{n+1}) = \left[ \frac{\sum_{p=1}^{n} (I_p)(t_n) * e_{pi}(t_n)}{\sum_{p=1}^{n} e_{pi}(t_n)} \right]$$

It should be noted that when a particular resource becomes the target of different coordinated attacks launched from different sources, for the computation of the incident strengths, the individual incidents that occur for the resource are taken into consideration to compute overall impact of the incidents on the resource, irrespective of their context. The fusion model combines the strengths of the different incidents activated for a particular resource in order to measure the extent of its incident association.

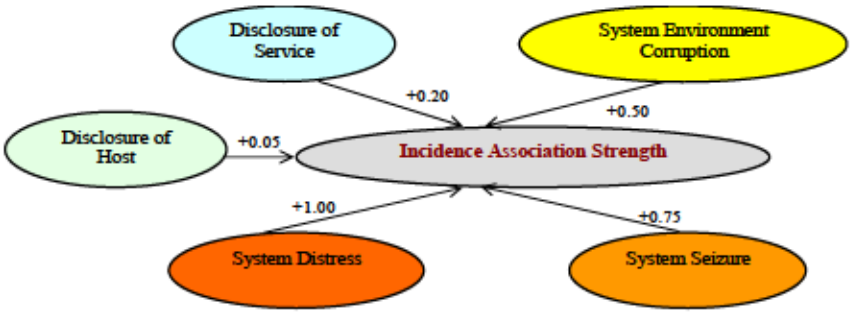


Figure 3.8 Combining Evidence of Security Incidents

Figure 3.8 shows how the evidence of different incidents activated for a resource contribute to the overall incident association of the resource with different impacts. The degree of impact depends on the nature of the incident and the designated security policy. As information assurance needs are different in different organizations, fusion models must be sensitive to a system's/installation's/site's security goals and polices. Therefore, the degrees of impacts by the incidents should be determined from the security policy. At any time, the *Incident Association Strength* (IAS) of a particular resource collectively represents the effects of all the security incidents activated for the resource at that time.

Therefore, the IAS of a resource $R_i$ at time $t_{n+1}$ for each contributing incidents $I_k$ with impact $e_{ki}$, can be represented as the following:

$$IAS(R_i)(t_{n+1}) = \left[ \frac{\sum_{k=1}^{n} (I_k)(t_n) * e_{ki}(t_n)}{\sum_{k=1}^{n} e_{ki}(t_n)} \right]$$

With abstract incident modeling, the alert fusion model correlates alerts to find incidents in a coordinated attack scenario. Along with incidents found for each resource, their Incident Association Strengths (IAS)s are also reported. It should be pointed out that IAS can be considered a confidence score given by the fusion model to represent the degree of concern for a particular resource's involvement in correlated security incidents resulting from multi-staged attacks.

### *3.2.3 Situation Assessment*

Situation assessment is conducted for a better understanding of the security health of protected resources in the distributed network. A condensed view of the security status of the network presented in terms of overall degree of concerns for the protected resources can help to prevent information overload and aid in improved situational awareness.

For misuse sensors that are able to indicate the specific nature of the attacks, the overall degree of concern for each resource in the network jointly depends on the resource's involvement in the security incidents resulting from multi-staged attacks and in common attack patterns. As described in sections 3.2.1 and 3.2.2, abstract alert correlation reports incident association strength (IAS) as the degree of concern for a

resource's involvement in incident association, while multi-level alert clustering reports cluster association strength (CAS) as the degree of concern for a resource's involvement in cluster association. Therefore, it makes sense to combine these two results of alert association for overall situation assessment for the protected resource. In this regard, a *dynamic fusion* approach is proposed with the following general characteristics:

- It is possibilistic in nature. From fuzzy sets and possibility theory, the fusion process adapts fuzzy information combination operators to fuse multiple fuzzy inputs to derive final output.
- It is dynamic. The fusion process is context dependent [3], i.e., while fusing information, it behaves in accordance with some contextual[21] information about the inputs.

In the dynamic fusion approach for misuse situation assessment, the incident and cluster association strengths reported by the alert fusion model (which take their values from the closed interval [0,1]) are considered measures of concern for the resources in related security situations. These inputs are represented using fuzzy sets and possibility theory such that possibilistic combination operators can be applied. There are three major steps in the dynamic fusion process - shown in Figure 3.9:

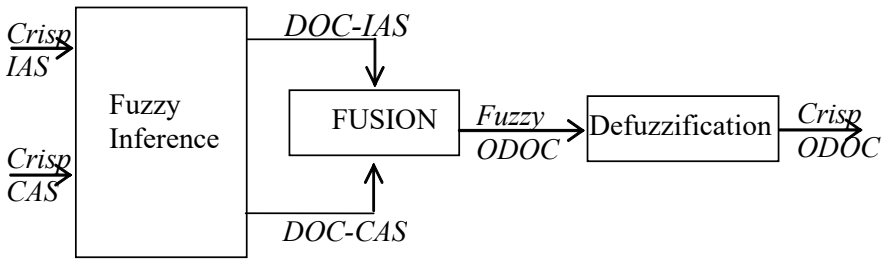


Figure 3.9 Dynamic Fusion Process

[21] Contextual information is information external to the values of the information to be combined. It can be some global knowledge about the information or measure on the sources to be fused [3].

The first step is the fuzzy inference process for the possibility distribution generation of the inputs where the crisp inputs denoting IAS and CAS are transformed into possibility distributions. The inputs are fuzzified and aggregated according to the Mamdani model [62]. At the end of the fuzzy inference step, possibility distributions of Degree of Concerns (DOC)s as related by IAS and CAS, are obtained (DOC-IAS and DOC-CAS). Figure 3.10 shows the possibility distribution of the fuzzy variable DOC.

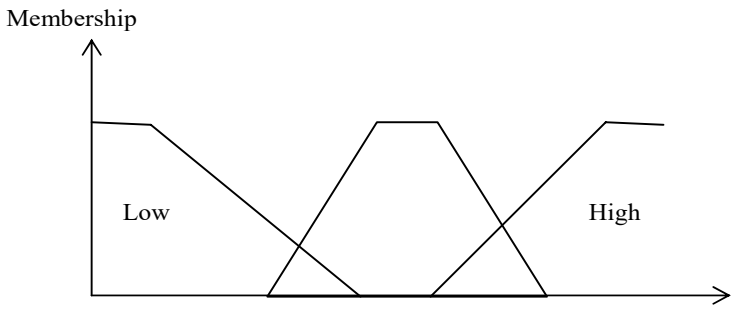


Figure 3.10 Complete Term Set for Fuzzy Variable Degree of Concern

The actual fusion takes place in the next step (Figure 3.9) where the fuzzy DOCs are combined into a new possibility distribution for overall degree of concern (ODOC). Therefore,

$$\Pi_O(\omega) = \Pi_i(\omega) \otimes \Pi_c(\omega)$$

where $\Pi_i$ represents the possibility distribution of the degree of concern conveyed by IAS (DOC-IAS), $\Pi_c$ represents the possibility distribution of the degree of concern conveyed by CAS (DOC-CAS), $\Pi_o$ represents the global possibility distribution of ODOC and $\otimes$ represent a possibilistic information combination operator.

In possibility theory, there are many choices for the ⊗ operator with different behavior properties [3]. Among them, we opted to use context dependent operators so that fusion is not only dependent on the inputs but also on some external context. As such context, agreement between the inputs is used. When two inputs totally agree, there is no conflict between them. If they don't, there can be weak or strong conflict between them depending on the degree of agreement. When the two inputs totally disagree, there is total conflict between them. Use of conflict in information fusion has been proposed by Dubois and Prade in [12] and discussed in [3, 46]. Using conflict (or degree of agreement) allows the fusion process to behave in one way when the inputs agree and in another way when they don't.

For dynamic fusion in misuse situation assessment, a new blend of possibilistic information combination operators is presented. Given two distributions $\Pi_i$ and $\Pi_c$, the global measure of conflict between the two can be designated as *(1-h),* where *h* is the height of the intersection of the possibility distributions and is given by [3, 12],

$$h(\Pi_i, \Pi_c) = sup_{\omega \in \Omega}(min(\Pi_i(\omega), \Pi_c(\omega)))$$

We refer to *h* as the consensus degree. The consensus degree between two inputs denotes the maximum extent of agreement between the two. When $h=0$, there is total conflict (i.e., no agreement) and when $h=1$ there is no conflict (i.e., total agreement). When $0<h<1$, there is partial conflict or agreement. As *h* decreases, conflict becomes stronger, and as *h* increases, conflict becomes weaker. The fusion process changes its behavior depending upon this consensus degree between the inputs; hence the term dynamic fusion. The changes in behavior for misuse situation assessment are as follows:

– When there is some agreement between the inputs (i.e., when $0 \leq \Pi_i(\omega) < h$ and $0 \leq \Pi_c(\omega) < h$), with both inputs within the consensus degree, additive behavior in fusion takes place. This is because in this case both alert correlation and alert clustering are supporting some concerns for the resource in question. Since agreement strengthens credibility, it makes sense to raise the sensitivity or extent of concern even more, depending on the inputs. In this regard, because of its characteristic additive behavior, the possibilistic fusion operator Hamacher Sum [64] is employed for combining the two inputs. The Hamacher Sum (HS) operator is defined below [64]:

$$HS(\Pi_i(\omega), \Pi_c(\omega)) = \frac{\Pi_i(\omega) + \Pi_c(\omega) - (2-\gamma)\Pi_i(\omega) * \Pi_c(\omega)}{1-(1-\gamma)\Pi_i(\omega) * \Pi_c(\omega)}, \gamma \geq 0$$

– When there is some disagreement between the inputs (i.e., when $0 \leq h < \Pi_i(\omega)$ or $0 \leq h < \Pi_c(\omega)$), with any input beyond the consensus degree, compromising behavior in fusion takes place. This is because conflict between the inputs essentially indicates that alert correlation and alert clustering are not supporting similar concern for the resource in question. It may be the case that one of them does not report any concern at all. However, then the one that does should not be ignored. Therefore, it makes sense to reach a compromise between the inputs. In this respect, the mean operator [3] is employed for averaging the concern supported by the inputs. The mean (MN) operator is defined as below:

$$MN(\Pi_i(\omega), \Pi_c(\omega)) = \frac{\Pi_i(\omega) + \Pi_c(\omega)}{2}$$

In order to incorporate changes in behavior of the fusion process in accordance with the consensus degree, conflict related rules from [12] are adapted to propose the following new fusion rule for the dynamic fusion process. According to the rule, initially two tasks are performed in parallel that constrict the choice of additive and compromised operators in accordance with the consensus degree (i.e., importance given to each is governed by *h*) and are described as below:

- $Constrained\_HS(\Pi_i(\omega),\Pi_c(\omega)) = min(HSN(\Pi_i(\omega),\Pi_c(\omega)),h(\Pi_i(\omega),\Pi_c(\omega)))$
    - This constricts the Hamacher Sum operator for additive behavior such that it gets weaker as consensus degree *(h)* decreases. It has no effect when there is total conflict or disagreement, i.e., $h=0$.
- $Constrained\_MN(\Pi_i(\omega),\Pi_c(\omega)) = max(MN(\Pi_i(\omega),\Pi_c(\omega)),h(\Pi_i(\omega),\Pi_c(\omega)))$
    - This constricts the Mean operator for compromising behavior such that it gets weaker as consensus degree *(h)* increases. It has no effect when there is no conflict or disagreement between inputs, i.e., $h=1$.

Finally, the fusion rule selects the appropriate combination operation based on the consensus degree as:

$$\forall\omega\in\Omega,\Pi_O(\omega) = \begin{cases} min(Constrained\_HS, Constrained\_MN) & \text{if } h < Constrained\_HS \\ max(Constrained\_HS, Constrained\_MN) & \text{if } Constrained\_MN > h \\ h & \text{otherwise} \end{cases}$$

- This rule selects a result based on the consensus degree. For inputs within consensus degree, additive behavior takes priority. For inputs beyond consensus degree, compromising behavior takes priority. At the level of consensus, the output equals $h$, i.e., maximum of the input values.

The final step in the dynamic fusion process for misuse situation assessment is to derive the final crisp output for the fuzzy Overall Degree of Concern (ODOC). In this respect, the centroid defuzzification method is employed as it ensures contribution of distributed data [11]. The centroid method calculates the weighted average of a fuzzy set [62] and is expressed as,

$$ODOC = \frac{\sum_i \Pi_o(\omega_i) \times x_i}{\sum_i \Pi_o(\omega_i)}$$

This overall degree of concern represents the quantitative assessment or confidence score given by the fusion model for overall security situation assessment of each protected resource in a distributed network using only misuse sensors.

When only anomaly sensors are used that are not able to indicate the specific nature of the attacks (a situation typical in high performance cluster environment), the overall degree of concern for each resource in the network primarily depends on the extent of the anomaly reported for the particular resource. Although anomaly sensors can detect known or unknown attacks as manifestations of anomaly, they often suffer from false positives where legitimate deviation from normalcy is also mistakenly flagged as an intrusion. Therefore, it makes sense to somehow substantiate the reports from anomaly sensors with some additional evidence in order to better understand the security situation. In this regard, we propose a new concept of *sensor corroboration* which makes use of *primary* and *secondary* sensors for complementary evidence support. Anomaly sensors are employed as primary sensors to monitor systems' security status. This type of primary sensor reports an anomaly as event-based evidence[22]. A set of secondary sensors are used to monitor different aspects of the system's (i.e., protected resource's) state. This type of secondary sensor reports system state attributes as state-based evidence[23], which can serve as complementary intrusion evidence to the event-based evidence or anomaly reported. Under normal conditions, the primary sensor monitors activities across the

[22] Event-based evidence refers to detection of possible intrusive actions [65].
[23] State-based evidence refers to observations of the effects of intrusions on system states [65].

system environment and reports alerts or event-based evidence of possible intrusions. The fusion model calls upon the secondary sensor only when the criticality of the situation dictates and/or there is need to substantiate the intrusion evidence (i.e., in this case, when any anomaly is reported by the primary sensor). The secondary sensor is used to provide evidence of alteration of system state attributes and can typically reside on an individual system, where it monitors and stores system attribute data locally in order to save communication overhead. Data is sent to the fusion model only on an "on-demand" basis. Data from secondary sensor is used to corroborate or challenge the primary sensor's reports and to evaluate the overall security situation. This type of fusion scheme should be particularly useful in resource constrained high performance cluster environments with a potential to save on communication overhead and resource utilization.

For anomaly situation assessment with primary and secondary sensors, the overall degree of concern for each resource in the network jointly depends on the extent of any event anomaly reported for the resource in a given time period and the alteration of any state attribute in the resource environment in that time period. An example of such system state attribute alteration can be change in the available memory of the resource.

The dynamic fusion process for anomaly situation assessment is similar to the one described for misuse situation assessment. The differences are in the parameters to be fused and the particular information combination operators used. This is denoted by,

$$\Pi_O(\omega) = \Pi_p(\omega) \otimes \Pi_s(\omega)$$

where $\Pi_p$ represents the possibility distribution of event anomaly reported by primary sensor, $\Pi_s$ represents the possibility distribution of the state attribute alteration reported

by secondary sensor, $\Pi_o$ represents the global possibility distribution of Overall Degree of Concern (ODOC) and $\otimes$ represents a possibilistic information combination operator. In anomaly situation assessment, the global measure of conflict between the two input distributions can be designated as *(1-h),* where *h* is the height of the intersection of the possibility distributions and is given by [3, 12],

$$h(\Pi_p, \Pi_s) = sup_{\omega \in \Omega}(min(\Pi_p(\omega), \Pi_s(\omega)))$$

Like misuse situation assessment, the fusion process for anomaly assessment changes its behavior depending upon the consensus degree *(h)* between the inputs, where the changes in behavior are as follows:

- When there is some agreement between the inputs (i.e., when $0 \leq \Pi_p(\omega) < h$ and $0 \leq \Pi_s(\omega) < h$), with both inputs within the consensus degree, disjunctive behavior in fusion takes place. This essentially means that the evidence from secondary sensor is supporting the report of anomalous situation from the primary sensor for the resource in question. In this regard, the possibilistic fusion operator Max [64] is employed for combining the two inputs. The Max (MX) operator is defined as below [64]:

$$MX(\Pi_p(\omega), \Pi_s(\omega)) = \text{MAX}(\Pi_p(\omega), \Pi_s(\omega))$$

- When there is some disagreement between the inputs (i.e., when $0 \leq h < \Pi_p(\omega)$ or $0 \leq h < \Pi_s(\omega)$), with any input beyond the consensus degree, compromising behavior in fusion takes place. This is because conflict between the inputs essentially indicates that the evidence from the secondary sensor is not supporting the evidence from the primary sensor for the resource in question. Therefore, it makes sense to reach a compromise between the inputs. In this respect, the weighted mean operator [3] is employed for combining the two inputs because of its compromising behavior. Weighted mean is used to attach more importance to the primary sensor's report than the secondary sensor's report. Here we attach three times more weight to the report from the primary sensor (i.e., event anomaly) than that from the secondary sensor (i.e., change in system attribute). The weighted mean (WMN) operator is defined as below:

$$WMN(\Pi_p(\omega), \Pi_s(\omega)) = \Pi_p(\omega) * 0.75 + \Pi_s(\omega) * 0.25$$

In order to incorporate changes in behavior of the fusion process in accordance with the consensus degree, conflict related rules from [12] are adapted to propose the new fusion rule for the dynamic fusion process for anomaly situation assessment. Initially two tasks are performed in parallel that constrict the choice of the disjunctive and the compromised operators in accordance with the consensus degree (i.e., importance given to each is governed by *h*) and are described as below:

- $Constrained\_MX\left(\Pi_p(\omega),\Pi_s(\omega)\right) = min\left(MX\left(\Pi_p(\omega),\Pi_s(\omega)\right),h\left(\Pi_p(\omega),\Pi_s(\omega)\right)\right)$
    - This constricts the Max operator for disjunctive behavior such that it gets weaker as consensus degree *(h)* decreases. It has no effect when there is total conflict or disagreement, i.e., *h=0*.
- $Constrained\_WMN\left(\Pi_p(\omega),\Pi_s(\omega)\right) = max\left(WMN\left(\Pi_p(\omega),\Pi_s(\omega)\right),h\left(\Pi_p(\omega),\Pi_s(\omega)\right)\right)$
    - This constricts the Weighted Mean operator for compromising behavior such that it gets weaker as consensus degree *(h)* increases. It has no effect when there is no conflict or disagreement between inputs, i.e., *h=1*.

Finally, the fusion rule selects the appropriate combination operation based on the consensus degree. If $\Pi_o$ represents the global possibility distribution of Overall Degree of Concern (ODOC), then the fusion rule follows as:

$$\forall \omega \in \Omega, \Pi_O(\omega) = \begin{cases} min(Constrained\_MX, Constrained\_WMN) & \text{if } h < Constrained\_MX \\ max(Constrained\_MX, Constrained\_WMN) & \text{if } Constrained\_WMN > h \\ h & \text{otherwise} \end{cases}$$

- This rule selects a result based on the consensus degree. For inputs within consensus degree, disjunctive behavior takes priority. For inputs beyond consensus degree, compromising behavior takes priority. At the level of consensus, the output equals *h*, i.e., maximum of the input values.

The final step in the dynamic fusion process is to derive a crisp output for the fuzzy Overall Degree of Concern with the centroid defuzzification method as discussed earlier. This Overall Degree of Concern represents the quantitative assessment or confidence score given by the fusion model for overall security situation assessment of each protected resource in a distributed network using only anomaly sensors.

It should be noted that to save computation, it makes sense to conduct dynamic fusion only when the report of event anomaly and the report of system state attribute alteration are not same. When they are, the same value is reported as the Overall Degree of Concern.

Dynamic fusion for misuse situation assessment primarily conducts decision fusion, where the results of the initial reasoning conducted on the sensor reported data (i.e., Degree of Concerns as related by IAS and CAS) are used to derive the final assessment. However, dynamic fusion for anomaly situation assessment conducts data fusion, where the sensor reported data (i.e., event-based evidence or event anomaly and state-based evidence or system state attribute alteration) are used directly to derive the final assessment.

Once the overall security situation is assessed for the protected resources in the network, a resource concern model can be used to relate the results of overall degree of concern to the security administrator. The greater the overall degree of concern, the more severe the security situation. The United States Department of Homeland Security (DHS) uses such a threat-based color coded system to communicate with public safety officials and the public at large, the current threat condition for the nation as posed by potential

terrorists (Figure 3.11). It is clear that an enormous number of factors must play into determining these threat levels to present a condensed view of the current threat condition. The alert fusion model uses a similar simplified scheme to relate the overall degree of concern to the security administrator with intuitive insight (Figure 3.12).


Figure 3.11 DHS Threat Model (Taken from [56])

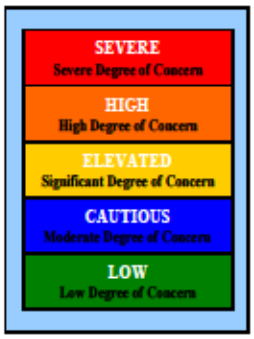


Figure 3.12 Resource Concern Model

This concern-based color coded condensed view of a protected system's security status reduces clutter for the security administrator and provides the security administrator with an evidence-based quantitative assessment for the system's security health that is representative of the degree of concern for its involvement in attack situations.

## 3.3 Summary

The unified alert fusion model provides a high-level reasoning capability beyond the low-level sensor abilities in order to deduce a condensed overall view of the security situation involving protected systems. The model has the following characteristics:

- it offers a possibilistic approach to alert fusion;
- it presents a unified architecture for alert fusion that combines alert prioritization, alert clustering, alert correlation for final situation assessment;
- it offers alert prioritization to reduce alert volume;
- it employs a multi-level alert clustering method to allow inexact matching between alert features;
- it employs an alert correlation method with abstract incident modeling to deal with scalability and uncertainty issues; and
- it offers situation assessment with dynamic fusion to allow sensitivity to conflict in results for both misuse and anomaly sensors.

# CHAPTER IV

# EXPERIMENTS AND RESULTS

In this chapter, we present the design, results and analyses of the experiments that were conducted to demonstrate the effectiveness of the alert fusion model. We developed prototype systems based on the models described in Chapter III in a UNIX environment. The alert fusion model was implemented in C++ with embedded SQL for data communication. The central repository of sensor alerts was implemented with a relational database with detailed drill-down capability. In the following sections, the detailed design of the experiments is presented along with the data sources that were used in the experiments. For each set of experiments conducted, evaluation criteria for the experiments are described, the effectiveness of the techniques is evaluated and the results measuring the performance of each technique are reported.

## 4.1 Experimental Design

The overall purpose of this research is to show that a unified alert fusion model, which combines alert prioritization, alert clustering and alert correlation in a single framework, is able to provide a security administrator with an overall condensed view of

the resources[24] in the network. This assessment can aid in improved understanding of the network's security health over sensor outputs with no fusion present. Therefore, the experiments conducted aim at evaluating the usefulness of the total approach, as well as each individual technique used in the approach.

A fusion system can perform a higher-level reasoning on alerts at the intra-sensor level (i.e., between alerts generated by a single sensor) and at the inter-sensor level (i.e., between alerts generated by different sensors). Therefore, this approach is evaluated both for intra-sensor data fusion (performed on a single sensor reports) and for inter-sensor data fusion (performed on multi-sensor report).

Since the alert fusion model's reasoning process is multi-dimensional, it is difficult to comprehensively evaluate the effectiveness of the whole approach. Individual experiments were designed with this difficulty in mind and have the following purposes:

- For misuse sensors,
    - Evaluate the alert fusion model's ability to *prioritize* low-level alerts in sensor alert reports.
    - Evaluate the alert fusion model's ability to *correlate* low-level alerts in sensor alert reports.
    - Evaluate the alert fusion model's ability to *cluster* low-level alerts in sensor alert reports.
    - Evaluate the alert fusion model's ability to *combine the results* of alert prioritization, alert clustering, and alert correlation to provide the security administrator with an overall condensed view of the system.

[24] For the purpose of presenting of results, throughout the rest of this chapter, we refer to resource as a host in a dedicated network of computers.

- For anomaly sensors,
    - Evaluate the alert fusion model's ability to *combine the data* reported by primary sensor and secondary sensor to provide the security administrator with an overall condensed view of the system.

## 4.2 Experimental Setup

For experimentation with misuse sensors, MIT Lincoln's Lab's DARPA[25] (LLD) 2000 Intrusion Detection Evaluation (IDEVAL) Scenario Specific dataset [34] was used as the test data because it is a well known benchmark dataset that contains simulated multi-staged attack scenarios in a protected environment. We used this dataset for which ground truth is known because it allowed us to assess the success of our experiments and compare our experimental results to work by other researchers in this area who have also used this dataset to report their results. Also the fact that the ground truth required for validation purposes cannot be known for real world traffic, has inspired us to use this simulated attack traffic.

In the LLD experiment, the attack traffic includes a series of attacks carried out over multiple networks and audit sessions by an attacker who probes hosts in the network, successfully breaks into some of them to prepare for and finally launch Distributed Denial of Service (DDoS) attacks against an off-site government website. Figure 4.1 shows the service plot for the intrusion scenario.

[25] Defense Advanced Research Projects Agency.

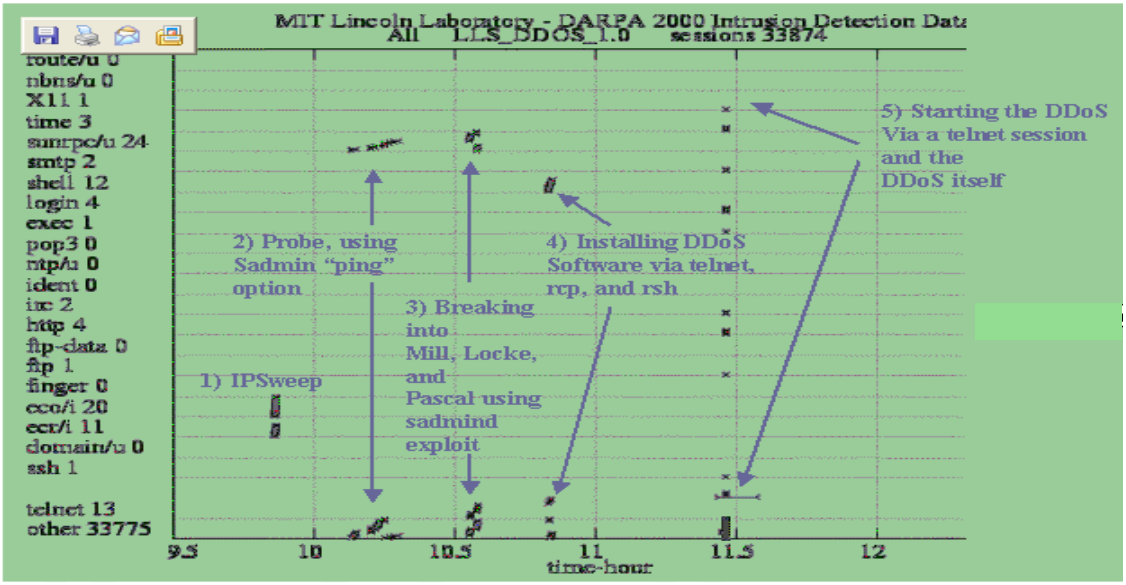


Figure 4.1 Service Plot for Lincoln Lab's DARPA 2000 Intrusion Scenario (Taken from [35])

Here, attack experiments were conducted over three segments of a simulation network: a network inside an Air Force base, an internet outside an Air Force base and the demilitarized zone (DMZ) that connects the outside network to the inside network [34]. There are two attack scenarios in the LLD attack traffic:

- one scenario includes DDoS attacks carried out by a novice attacker (LLDOS 1.0) who compromises three hosts individually to launch the attack against an outside host;
- another scenario includes DDoS attacks carried out by a more sophisticated attacker (LLDOS 2.0.2) who compromises one host and then fans out from it.

In general, attacks in LLDOS 2.0.2 are stealthier that those in LLDOS 1.0. Overall, there are four *tcpdump* files containing the attack traffic:

1. LLDOS 1.0 Inside Zone;
2. LLDOS 1.0 DMZ;
3. LLDOS 2.0.2 Inside Zone; and
4. LLDOS 2.0.2 DMZ.

The LLD website also provides a list of all the hosts in the three segments of the evaluation network [34].

For evaluating our fusion technique, the LLD attack traffic itself was not sufficient because we required sensor alert reports that would result from monitoring this attack traffic. Therefore, for the purpose of our research, we needed to generate these sensor alert reports as part of the research. In this regard, two intrusion detection systems (IDS)s or sensors, RealSecure (Version 7.0) [21] (a commercial signature based network sensor) and Snort (Version 2.3.3) [47] (an open source lightweight signature based network sensor) were used. Apart from the fact that these sensors are among the most widely used sensors today, RealSecure was selected because other researchers have also used this IDS for similar purposes [38] and Snort was selected because it is a freely available IDS.

To generate the sensor alert reports, Snort was configured to execute in full coverage (with all available attack signatures active) and installed in the security lab of the Center of Computer Security Research (CCSR), Mississippi State University (MSU), to monitor the simulated network traffic containing the LLD attacks. Since RealSecure is not designed to monitor offline *tcpdump* data, the *tcpdump* files had to be replayed in a live network using *tcpreplay*, a *tcpdump* file utility program offered by Open Source Technology Group [42].

We elected to execute RealSecure with the "Attack Detector" policy, instead of the "Attacks and Audits" policy (which is equivalent to executing Snort with all the default rules), for the following reasons:

- Executing RealSecure and Snort with an equivalent policy generated almost the same sets of alerts from both sensors since they both sniffed the same attack traffic. Consequently, we found that the alert fusion model reported the same results by analyzing the sensor reports individually. Apart from showing that the fusion model performs consistently, it is not interesting to compare and contrast the results. Differences in sensor scope coverage resulted in different sensor reports and we wanted to evaluate the fusion model's performance in detecting these differences.

- In real situations, it is likely that one will need to make use of sensors with different coverage. From a security administrator's point of view, provided with two identical sensors, it makes more sense to use one of them with full coverage and one with focused coverage. This is because full coverage generates a very large amount of alerts in sensor report (for example, RealSecure generated more than 39K alerts for the LLDOS 1.0 inside zone attack traffic alone), which not only included clear attacks but also include all audits for any kind of notable activities. On the other hand, for a better understanding of the big picture, sometimes it is beneficial to analyze all activities to trace the malicious ones to their roots or to link them together.

In addition to these generated sensor alert reports, for evaluating intra-sensor fusion, an additional sensor alert report was used in the experiments that has been made available by researchers at North Carolina State University (NCSU) as part of the TIAA (a Toolkit for Intrusion Alert Analysis) project [39]. This sensor report consisted of alerts generated by RealSecure (Version 6.0), executed with the "Maximum Coverage" policy (which is equivalent to the "Attacks and Audits" policy of Version 7.0) against the LLD attack traffic. We used this sensor report to campare our work with other researchers who have also used this dataset for evaluating their work [38, 63].

Evaluating inter-sensor fusion, i.e., evaluating the alert fusion model's performance in building an overall security view by analyzing integrated alerts reported by multiple sensors (in our case, RealSecure and Snort), involved multi-sensor data generation consisting of:

- simulation of attacks in a live network with *tcpreplay* and LLD attack traffic;
- installation of both sensors such that they would monitor the same traffic simultaneously and generate alerts independently; and finally
- integration of the individual sensor alert reports.

Both RealSecure and Snort were installed in the same host to monitor the LLD attack traffic. To integrate the sensor alerts, the following alert features were extracted from the independent sensor reports: Source, Target, Time, and Attack name. An additional feature identified the sensor and the individual alerts uniquely.

For experimentation, the generalization hierarchy shown in Figure 3.2 of Chapter III, was used to generalize the attack names in the sensor alert reports into abstract alert types. Since the sensors (Real Secure and Snort) used in the experiments were both signature-based or misuse sensors, alert abstraction was limited up to level 2 of the generalization hierarchy. The low-level alerts reported by RealSecure were generalized with the help of attack signature descriptions provided by ISS, Inc.'s X-Force database, a very comprehensive threats and vulnerabilities database (http://xforce.iss.net/), and generalization of the Snort alerts were conducted using attack signature descriptions provided by Sourcefire, Inc. (http://www.snort.org/). In addition, security experts were consulted for their comments/suggestions on the generalization. Please refer to Appendix B for the complete categorization of the attack names reported by RealSecure and Snort.

For the sensor corroboration experiment with the use of an anomaly sensor, the initial plan was to conduct a real-time experiment in the experimental high performance computing cluster within the Department of Computer Science and Engineering (CSE) at MSU. The cluster consisted of multiple internal nodes interconnected through both Ethernet and Gigabit switches. In this environment, we specially wanted to investigate our alert fusion model's ability to corroborate evidence between primary and secondary sensors to provide better assessment of attack situations, where attacks specifically designed for and applicable in the Linux cluster environment would be executed. These attacks [55], developed by the CCSR, were designed to create specific confidentiality/integrity/availability issues within a cluster and have been successfully tested for the MPI/PRO and C environments. In other work within the CCSR [13, 31], multiple anomaly detection techniques were developed to capture and report anomalies in the function/system calls generated by these cluster attacks. The primary experimental plan was that, while these anomaly sensors would serve as primary sensors in the cluster nodes, for evidence corroboration a secondary sensor would be used along with the primary sensors to demonstrate how the dynamic fusion approach could help in overall security situation assessment in the cluster environment. The task of the secondary sensor would be served by a performance monitoring tool, Ganglia [32]. Ganglia is an open-source project that grew out of research at the University of California, Berkeley, and has gained popularity as a scalable distributed monitoring system for high-performance computing systems.

At first, we expected to use multiple anomaly sensors working simultaneously to report alerts against multiple cluster attacks. However, certain difficulties forced us to revise this initial plan to use a single anomaly sensor monitoring a single cluster attack. First, we could not use multiple anomaly sensors because these sensors [13, 31], which would capture anomalies with function and system calls, were not developed to work simultaneously on the same host due to implementation issues. Secondly, the cluster attacks [55], which were specifically designed to cause anomalies at the function and system call levels, in most cases (except one) would not generate effects such that the effects were detectable by performance monitoring tools, other than using the specialized anomaly detection techniques [13, 31]. Therefore, we decided to continue the experiment with a single primary sensor monitoring against a single attack (memory allocation attack [55]), which would cause anomalies both at the function call level and the system attribute level (particularly with memory). This would enable the attack to be detected both by the primary sensor and the secondary sensor (Ganglia) such that we would be able to combine their reports to corroborate evidence for final situation assessment. Up till now, individually we could demonstrate that executing the memory allocation attack caused anomalies at the function level detectable by an anomaly sensor [13], and exhaustion of memory detectable by a secondary sensor, i.e., Ganglia. However, unfortunately due to implementation issues, we were unable to monitor the attack in real-time simultaneously with both the primary (anomaly) and secondary (Ganglia) sensors.

Since we were not able to attain real-time data, for our research in this dissertation, we had to conduct experiment on synthetic data representing event and state based evidence reported by primary and secondary sensors to evaluate the sensor corroboration experiment.

## 4.3 Experimental Results

For the misuse sensors (RealSecure and Snort), several experiments were conducted to evaluate the performance of the alert fusion model in conducting intra- and inter-sensor fusion - combining alert prioritization, abstract alert correlation and multi-level alert clustering in a unified framework. Each of these different techniques was evaluated against all four of the sensor reports (RealSecure-NCSU sensor report, RealSecure-MSU sensor report, Snort-MSU sensor report and MultiSensor-MSU report with RealSecure and Snort) generated from the LLD attack traffic, where each of the sensor reports consisted of four individual datasets.

The initial set of experiments was conducted for alert prioritization. We found that conducting alert prioritization reduced alert volume substantially. After alert prioritization, the prioritized sensor alerts were used individually for conducting abstract alert correlation and multi-level alert clustering. We found that while abstract alert correlation identified alerts associated with multi-staged attacks, multi-level alert clustering identified alerts associated with common attack patterns. The results of alert correlation and alert clustering were finally combined in the last set of experiments for overall situation assessment. We found that situation assessment for misuse sensors effectively combined the results of the hosts' involvement in multi-staged security

incidents and in common alert clusters. In our final experiment, we found that situation assessment for anomaly sensor effectively combined reports from primary and secondary sensors to corroborate evidence of attack for final situation assessment. The following details these different experiments that individually focus on the different techniques - with their specific objectives, evaluation methods, results and analysis.

### *4.3.1 Alert Prioritization Experiment*

*Objective*

The purpose of this experiment was to investigate the alert fusion model's ability to prioritize alerts by considering the relative importance of the information contained within each alert as dictated by a designated security policy.

The main objective of prioritization is to extract "interesting" data from the sensor reports for more efficient analysis of the data. In our case, *source/target criticality* and *attack criticality* jointly determined the priority of the alerts for the misuse sensor alerts. Appendix C shows a complete listing of criticality indexes of source/target and attacks used in this experiment.

*Evaluation*

After prioritization, the prioritized alerts were manually scrutinized to investigate if prioritization was performed correctly. Low priority alerts, whose priority values fell below a threshold (in our case, it was 0.10), were filtered or excluded from further analysis of data (i.e., alert correlation and alert clustering). The main advantage of

filtering low priority alerts is alert reduction, which also aids in reducing false positives in sensor alert reports. To measure the effectiveness of alert prioritization in terms of reducing alert volume, we used the following evaluation metric:

- Alert Reduction Rate (ARR): is measured by the ratio of the number of alerts reported by the fusion model after prioritization, to the total number of actual alerts reported by the low-level sensor(s) before prioritization.

*Results and Analysis (R&A)*

The experiment was conducted on four sensor alert reports - RealSecure-NCSU, RealSecure-MSU, Snort-MSU and MultiSensor-MSU individually. Figures 4.2, 4.3, 4.4 and 4.5 show notable alert reduction achieved by the alert fusion model with alert prioritization.

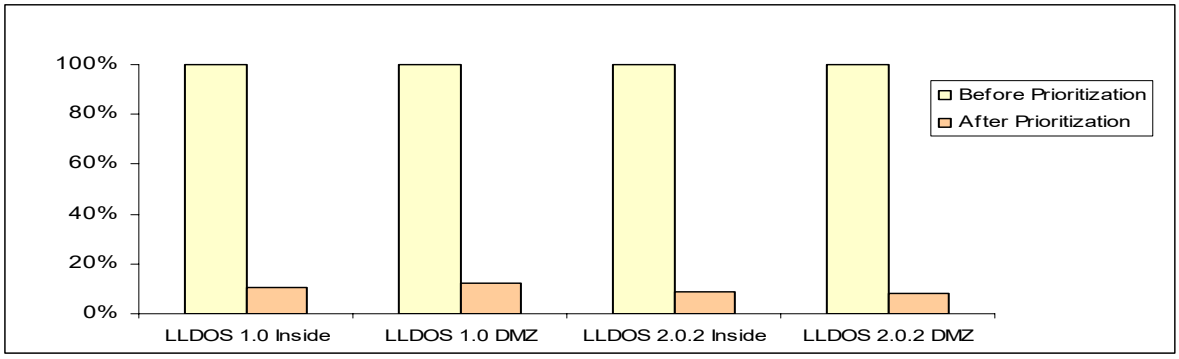


Figure 4.2 Alert Reduction with Prioritization for the RealSecure-NCSU Sensor Report

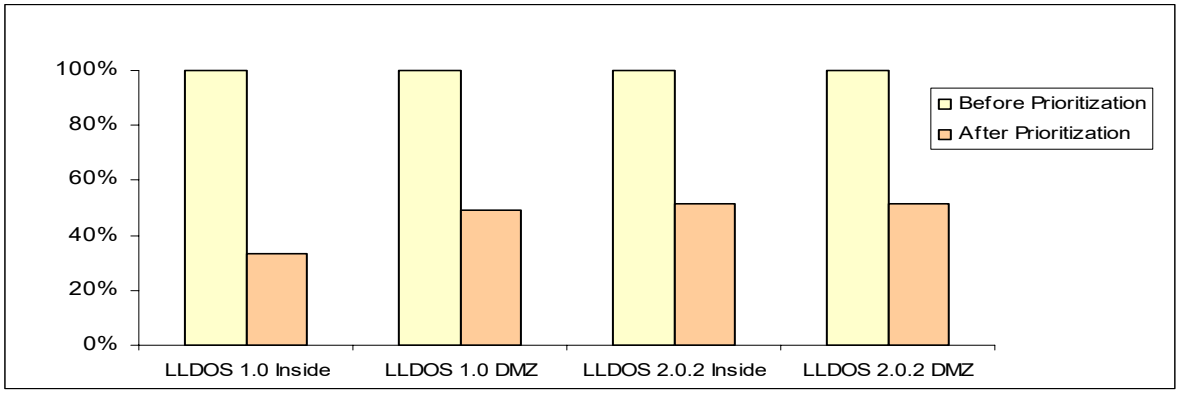


Figure 4.3 Alert Reduction with Prioritization for the RealSecure-MSU Sensor Report

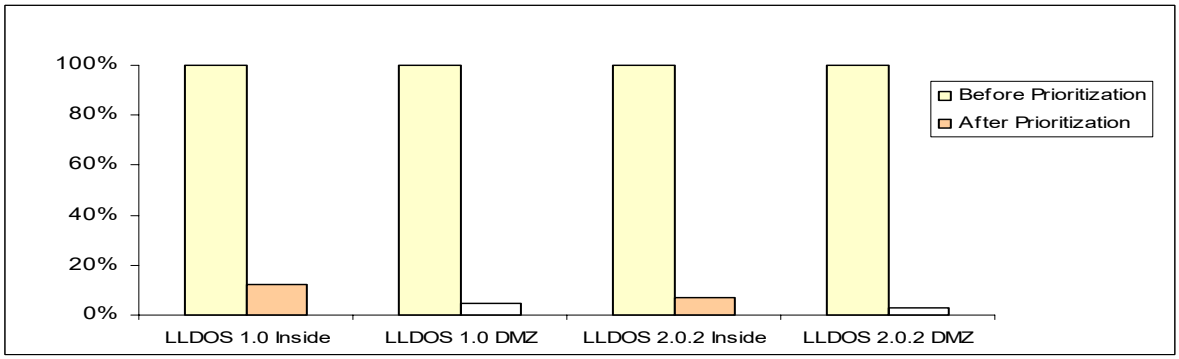


Figure 4.4 Alert Reduction with Prioritization for the Snort-MSU Sensor Report

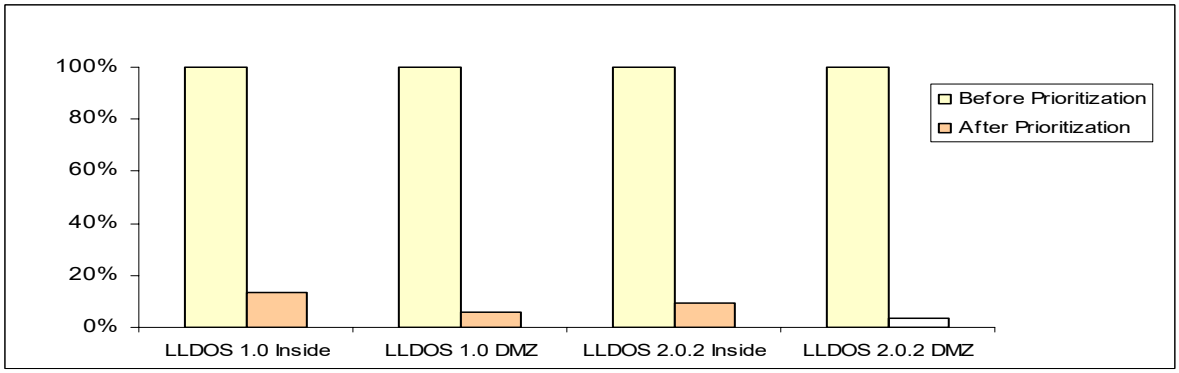


Figure 4.5 Alert Reduction with Prioritization for the MultiSensor-MSU Report

Although all of the figures show alert reduction with prioritization, we can see that the results are best for the sensor reports with large numbers of alerts (i.e., Snort-MSU and MultiSensor-MSU reports). When Snort was executed with application of all misuse signatures (both attacks and informational rules), the DMZ datasets showed a large number of *ICMP_Redirect_Host* alerts (2159 in LLDOS 1.0 DMZ dataset and 1479 in LLDOS 2.0.2 DMZ dataset) that came from the DMZ host *loud*: 172.016.114.001 directed to other DMZ hosts. These alerts were generated as a result of the host *loud* sending messages to other DMZ hosts about the existence of the *firewall*: 172.16.114.002. As a result, both Snort-MSU and MultiSensor-MSU reports contained this large numbers of false positives. These alerts were considered low priority alerts by the fusion model for two reasons. According to the generalization hierarchy shown in Figure 3.2 of Chapter III, *ICMP_Redirect_Host* alerts generalize to the *Policy_Compliance_Suspicious* category, which has a low information criticality (0.05). Also, as the alerts were generated between two DMZ hosts, the source/target criticality was considered moderate (0.75). Taking both factors into account, the criticality of such alerts was derived to be very low (0.0375), falling well below the priority threshold (i.e., 0.10). As a result, these large volumes of false positives were filtered out from further analysis.

Among all the sensor reports, the alert prioritization of the RealSecure-MSU sensor report resulted in the least improvement in terms of alert reduction. This makes sense because in this case, RealSecure was executed with the "Attack Detector" policy,

which generates alerts for real threats only, and therefore, since a majority of the alerts in the sensor report were genuine, there was less opportunity for filtering out false positives. Hence, we found the least alert reduction for this particular sensor report.

*Summary of Alert Prioritization Experiment*

In the alert prioritization experiment, we found that the fusion model was able to prioritize alerts based on source/target and attack criticality for the misuse sensor reports. Filtering out low priority alerts resulted in notable alert reduction. Also, as we find in later experiments, excluding the low priority alerts from further analysis did not hamper the identification of the victim hosts in the alert correlation and the alert clustering experiments.

### *4.3.2 Alert Correlation Experiment*

*Objective*

The purpose of the alert correlation experiment was to investigate how accurately the alert fusion model was able to correlate alerts that are part of coordinated attacks to determine a host's involvement in multi-staged attacks. The experiment was conducted on four sensor alert reports - RealSecure-NCSU, RealSecure-MSU, Snort-MSU and MultiSensor-MSU individually. While the individual sensor reports were used to evaluate and compare the alert fusion model's alert correlation performance for intra-sensor fusion, the multisensor report was used to evaluate its performance for inter-sensor fusion.

*Evaluation*

Evaluating a high-level reasoning process, like alert correlation with abstract incident modeling (abstract alert correlation), is not trivial as it involves many subjective and qualitative factors. To show the alert correlation capability of the fusion model, the following correlation performance metrics are used in this dissertation, as suggested by Qin and Lee [45]:

- True Causality Rate (TCR): is measured by the ratio of the number of correctly correlated alerts[26] for a scenario to the total number of actual causal relationships[27] that are involved in the scenario.
- False Causality Rate (FCR): is measured by the ratio of the number of incorrectly correlated alerts for a scenario to the total number of correlated alerts reported for the scenario.

It should be pointed out that TCR essentially measures the detection rate for alert correlation and can be used as an indicator of how completely the alert fusion model is able to correlate alerts. FCR measures the false positive rate for alert correlation and provides insight into how correctly the alert fusion model is able to correlate alerts.

In addition to these metrics, since abstract alert correlation also aids in alert reduction by reporting correlated alerts, results are also shown for alert reduction using the alert reduction metric described in section 4.3.1. The LLD documentation for the attacks and the low-level sensor alert reports were used to determine the number of causal relationships in the data [35].

[26] Correlated alerts: Alerts that are reported by the fusion model to be part of coordinated attacks.
[27] Causal relationships: Alert data that are part of coordinated attacks against target hosts.

*Results and Analysis (R&A)*

The results and analysis of the alert correlation experiment is presented below. It should be noted that alert correlation is performed on data filtered with alert prioritization.

*R&A for RealSecure-NCSU Sensor Report*

In case of the LLDOS 1.0 Inside Zone dataset, the only hosts that the alert fusion model reported under attack were the three victim hosts (*mill*: 172.016.115.020, *pascal*: 172.016.112.050 and *locke*: 172.016.112.010), which the attacker compromised individually and then used to launch the DDoS attack. The chart in Figure 4.6 shows the DDoS attack scenario represented by the alerts that were correlated by the fusion model from analyzing the LLDOS 1.0 Inside Zone dataset. Four specific security incidents[28] identified by the fusion model represent four distinct phases of the attacks:

*Phase 1.* In this early phase, probing activities were conducted to discover services running on the hosts, which resulted in a *Discloser_of_Service* (*DSV*) incident. (Here the attacker probed the hosts with *sadmind ping* to detect which hosts had the *sadmind* service running.)

*Phase 2.* In this phase, exploitation attacks were executed that resulted in a *System_Environment_Corruption* (*SEC*) incident. (Here the attacker exploited the vulnerability associated with the *sadmind* service to gain root access into the victim hosts.)

*Phase 3.* In this phase, remote-to-root activities were carried out that resulted in a *System_Seizure* (*SSZ*) incident. (Here the attacker uploaded necessary files for installing *mstream* software on the compromised hosts via *telnet* and *rsh*.)

*Phase 4.* In this phase, attack tools were installed, which resulted in a *System_Distress* (*SDT*) incident. (Here the attacker installed *Trojan mstream* DDoS software to carry out the DDoS attacks from the victim hosts.)

[28] Please refer to section 3.2.2.2 of Chapter III for detail description of these incidents.

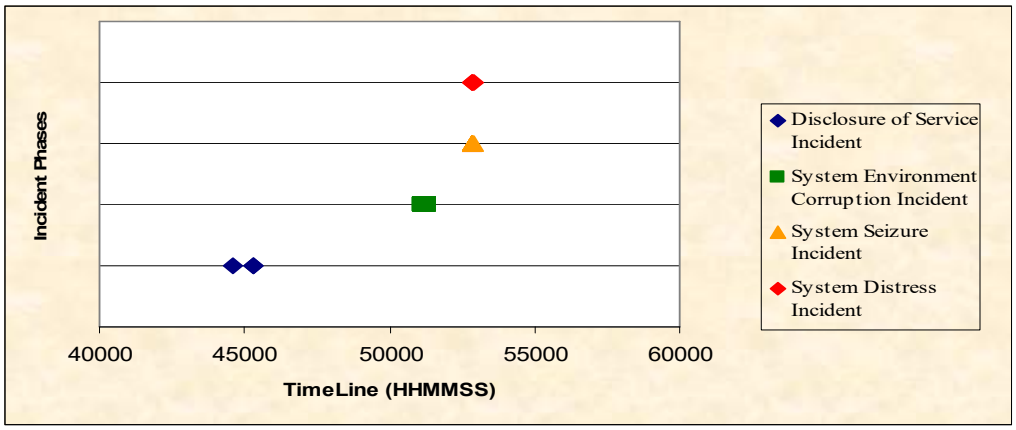


Figure 4.6 Correlated Alerts depicting the Attack Scenario for RealSecure-NCSU Sensor Report

According to the LLD documentation [34], there are two additional phases in the coordinated attacks, which were not reported by the alert fusion model. In the LLDOS 1.0 attack scenario, the attacker initially conducted an *IPSweep* of the network from a remote site. Therefore, phase 1 should actually indicate a *Discloser_of_Host* (*DHS*) incident. Since RealSecure did not generate an alert in this case, the fusion model was unable to detect this initial phase. Ning et al. conducted alert correlation and reported experimental results on the same datasets using hyper-alert correlation graphs [38]. In their paper, the authors reported this same problem during their experiments with the LLD data. This highlights the fact that effectiveness of any high-level analysis of sensor data is largely dependent on the quality of the data itself. The final phase of the DDoS attack concerns launching of the *Stream_DoS* attack. Although RealSecure reports this attack by generating *Stream_DoS* alert, the fusion model did not correlate this alert. The reason is that being resource-centric (as described in section 3.2 of Chapter III), the fusion model concentrates on communications to/from legitimate hosts in the network.

Since this final DDoS attack was launched from a spoofed IP address and was targeted for a host outside the network, the fusion model did not analyze the corresponding communication. However, this did not severely affect the incident situation awareness for the network because the critically significant *System_Distress* incident, which activated because of evidence of illegal DDoS tool installation, inherently raised the degree of concern for the host in question highly.

Table 4.1 denotes the abstract alert correlation results on the RealSecure-NCSU sensor report in terms of the metrics described earlier. It should be noted that this only concerns the correlated alerts reported by the fusion model. The abbreviations used are the following:

SA: *Sensor reported Alerts*, CR: *Causal Relationships*, CA: *Correlated Alerts*,
CCA: *Correctly Correlated Alerts*, ICA: *Incorrectly Correlated Alerts*,
MA: *Missed Alerts*, TCR: *True Causality Rate*, FCR: *False Causality Rate*.

Table 4.1 Alert Correlation Performance for the RealSecure-NCSU Sensor Report

| Dataset | SA | CR= CCA +MA | CA= CCA+ ICA | CCA | ICA | MA= CR- CA | TCR= CCA/CR | FCR= ICA/CA |
|---|---|---|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 922 | 44 | 44 | 42 | 2 | 2 | 95.45% | 4.54% |
| LLDOS 1.0 DMZ | 891 | 57 | 59 | 56 | 3 | 1 | 98.24% | 5.08% |
| LLDOS 2.0.2 Inside Zone | 494 | 20 | 13 | 13 | 0 | 7 | 65.0% | 0% |
| LLDOS 2.0.2 DMZ | 430 | 8 | 5 | 5 | 0 | 3 | 62.5% | 0% |

The column titled MA of Table 4.1 shows the number of alerts missed by the fusion model for each of the datasets. It can be seen that True Causality Rates (TCR)s for the

fusion model decrease with the number of alerts missed (i.e., alerts that were supposed to be correlated but were not). The following explains why the fusion model missed those alerts.

For every established telnet session, RealSecure (Version 6.0) generates three alerts: *Telnet_Terminal_Type* (*TTT*), *Telnet_Env_All* (*TEA*) *and Telnet_XDisplay* (*TXD*). In the fusion model, a *TTT* alert would activate the *Policy_Compliance_Notification*[29] CEvent (because this particular alert only notifies the initiation of a telnet session with the reported terminal type) and thus do not generate any incident that is part of a coordinated attack as described by the abstract incident model. Therefore the fusion model would not correlate the TTT alert. On the other hand, since both *TEA* and *TXD* alerts relate to the *Active_Communication* CEvent (because they denote actual execution of a remote session), the fusion model would be able to correlate these alerts as part of the attack scenario. This is the same reason for being able to correlate the alert *FTP_Put (FPT)*. The other two ftp session related alerts are *FTP_Pass* (*FPS*) and *FTP_User* (*FUS*) and because these alerts indicate initiation of an ftp session with reported password and user name and hence activate *Policy_Compliance_Notification* CEvent, the fusion model would not correlate them.

In the case of LLDOS 1.0 Inside Zone dataset, the number of missed alerts by the fusion model was two, which included one for *Stream_DoS* and one for *TTT*. For LLDOS 1.0 DMZ dataset, the one missed alert was the same *TTT* alert as in LLDOS 1.0 Inside

[29] It should be noted that none of the *Policy_Compliance* CEvents (i.e., *Policy_Compliance_Notifucation, Policy_Compliance_Suspicious, and Policy_Compliance_Informational)* are considered to be part of a coordinated attack, as described by the abstract incident model shown in Figure 3.7 of Chapter III.

Zone dataset. For LLDOS 2.0.2 Inside Zone dataset, the number of missed alerts was seven (two *TTT* alerts for two telnet sessions, two *FPS* and two *FUS* alerts for two ftp sessions, plus one more for *Stream_DoS*). For LLDOS 2.0.2 DMZ dataset, the number of missed alerts was three (one *TTT* alert for one telnet session and one *FPS* and one *FUS* alerts for one ftp session).

Table 4.1 shows that the TCRs reported are better for scenario one (LLDOS 1.0 Inside Zone & DMZ datasets) than for scenario two (LLDOS 2.0.2 Inside Zone & DMZ datasets). This is because of the lack of representative data for the attacks and the presence of multiple telnet and ftp sessions in the sensor alert report for scenario two.

The column titled ICA of Table 4.1 shows the number of alerts incorrectly correlated by the fusion model for each of the datasets. It can be seen that False Causality Rates (FCR)s for the fusion model increase with the number of alerts incorrectly correlated (i.e., alerts that were not supposed to be correlated but were). The following explains why the fusion model correlated those alerts.

The fusion model incorrectly correlated two alerts (*FTP_Syst* and *Email_Almail_Overflow)* in the case of LLDOS 1.0 Inside Zone dataset, generated for the inside host *crow:* 172.016.113.148. The reason for this correlation is that *Email_Almail_Overflow* alert activated a *System_Environment_Corruption* incident that occurred following a *Disclosure_of_Service* incident (activated due to *FTP_Syst* alert) - all initiated from the same source. Therefore as two sequential incidents in a multi-staged attack (as shown in Figure 3.7 of Chapter III) got activated, and the contributing alerts were correlated by the fusion model. In LLDOS 1.0 DMZ dataset, in addition to these

same two false positives as in LLDOS 1.0 Inside Zone dataset, there was an additional one for the alert *UDP_Port_Scan* (which activated a *Disclosure_of_Service* incident) for the host *pascal:* 172.016.112.050. While correlation of these alerts was justifiable, we count them as false positives since they are not mentioned in the LLD documentation.

As mentioned before, researchers in NCSU (Ning et al.) conducted alert correlation with hyper alert correlation graphs and reported their experiments results on this same sensor report in [37]. Apart from the primary difference in the technique used for correlation, there are also differences between us and NCSU in how we evaluate the performance of alert correlation and how we view ground truth in data. As pointed out by Ning et al., counting number of alerts or attacks is a subjective process that depends on how one views the attacks. Ning et al. refer to causal relations in sensor alert report as "related alerts" [37]. Our count of causal relations in the LLD attack traffic is slightly differently from NCSU (Table 4.2). Furthermore, Ning et al. define detection rate based on attacks rather than alerts (where "detection rate" is calculated as the ratio of the number of detected attacks to the number of observable attacks [37]). We compute detection rate or true causality rate based on alerts. In fact, our definition of detection rate is equivalent to NCSU's definition of "completeness measure", as found in [37]. Also, we use false causality rate to measure correctness of alert correlation and Ning et al. use "false alert rate" (defined as the complement of the ratio of the number of true alerts to the number of alerts [37]) for similar reason. Considering all these discrepancies, we refrain from any direct comparison of our results with that of NCSU. However, Table 4.2 shows ours and NCSU's results to be comparable.

Table 4.2 MSU and NCSU Alert Correlation Results for the RealSecure-NCSU Sensor Report

| Dataset | MSU Causal Relations | NCSU Related Alerts | MSU True Causality Rate | NCSU True Causality Rate * | NCSU Detection Rate** | MSU False Causality Rate | NCSU False Causality Rate * | NCSU False Alert Rate** |
|---|---|---|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 44 | 44 | 95.45% | 93.18% | 60.00% | 4.54% | 6.81% | 6.82% |
| LLDOS 1.0 DMZ | 57 | 57 | 98.24% | 94.74% | 56.18% | 5.08% | 5.26% | 5.26% |
| LLDOS 2.0.2 Inside Zone | 20 | 18 | 65.00% | 66.70% | 66.67% | 0% | 7.69% | 23.08% |
| LLDOS 2.0.2 DMZ | 8 | 8 | 62.50% | 62.50% | 42.86% | 0% | 0% | 40.00% |

* Calculated by MSU ** Reported by NCSU

The goal of abstract alert correlation is to report to the security administrator a list of the attacked resources that are involved in incident situations and the extent of their involvement in such situations by reporting their incident strengths and incident association strengths. Figure 4.7 is a snapshot of the incident situation discovered by the fusion model for a particular host *mill*[30]*:* 172.016.115.020, from analyzing the RealSecure-NCSU sensor report. In Figure 4.7, the x-axis shows the strength of the incidents that were activated for this host and y-axis shows the dataset the analysis was based upon.

[30] Throughout this dissertation, we frequently use examples involving this host because this is the only host, among all the victim hosts, for whom there were evidence of attacks found in all four of the datasets in the LLD attack traffic.

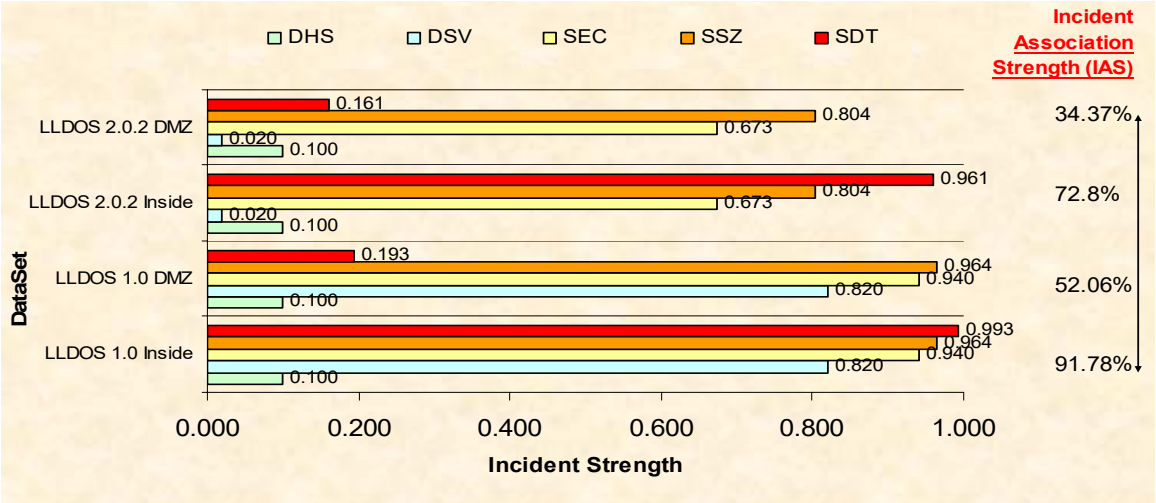


Figure 4.7 Alert Situation for Host *mil* for the RealSecure-NCSU Sensor Report

Figure 4.7 highlights how the evidence of the attacker's action (that causes the incident) and the pre-existing risk (that denotes the possibility of the incident occurring) jointly affect the strength of the security incident itself. For example, the *System_Environment_Corruption* (*SEC*) incident for *mill* has a high strength of 0.94 for scenario one (LLDOS 1.0 Inside Zone & DMZ datasets) and a moderate strength (0.673) for scenario two (LLDOS 2.0.2 Inside Zone & DMZ datasets). This is because for scenario two, the attacker used more sophisticated probing technique (*DNS HINFO queries*) that was not reported by RealSecure and therefore, no preceding EEvents or incidents were generated to place the host under the risk of an *SEC* incident. Since there was no such risk, later when evidence of a *SEC* incident surfaced, the incident did get activated but with less strength than compared to its activation with support of both evidence and risk. Figure 4.7 also shows the total incident association strengths for the

host for each of the datasets. It is found to be highest (91.78%) for LLDOS 1.0 Inside Zone dataset (because of the activation of all the incidents in the correlation chain except the least critical one, i.e., *Disclosure_of_Host* incident), and lowest (34.37%) for LLDOS 2.0.2 DMZ dataset (because of absence of the most detrimental incident, i.e., *System_Distress*).

Table 4.3. displays a list of the attacked hosts reported by the fusion model for all the datasets along with their incident association strengths (IAS)s from analyzing the RealSecure-NCSU sensor report. It should be noted that a significantly high IAS reported for a particular host indicates that multiple correlated incidents were found for the host, suggesting a major concern for the host's involvement in multi-staged attacks.

Table 4.3 Incident Association Assessment for the RealSecure-NCSU Sensor Report

| Dataset | Hosts | Incident Association Strength (IAS) |
|---|---|---|
| LLDOS 1.0 Inside Zone | 172.016.112.010 | 91.78% |
| | 172.016.112.050 | 91.78% |
| | 172.016.115.020 | 91.78% |
| | 172.016.113.148 | 27.0% |
| LLDOS 1.0 DMZ | 172.016.112.010 | 52.06% |
| | 172.016.112.050 | 52.06% |
| | 172.016.115.020 | 52.06% |
| | 172.016.113.148 | 27.0% |
| | 172.016.114.010 | 27.0% |
| | 172.016.114.020 | 27.0% |
| | 172.016.114.030 | 27.0% |
| LLDOS 2.0.2 Inside Zone | 172.016.112.050 | 72.8% |
| | 172.016.115.020 | 72.8% |
| LLDOS 2.0.2 DMZ | 172.016.115.020 | 34.37% |

The shaded rows show the hosts for which highest IAS was reported for each of the datasets and the listings in <BOLD> designate the hosts that were actually pursued by the attacker in the LLD experiments. There are four hosts (*plato:* 172.016.114.010, *smith:* 172.016.114.020, *solomon:* 172.016.114.030 and *crow:* 172.016.113.148) reported in Table 4.3 that were not listed as compromised according to the LLD documentation [34]. Although there is definite evidence that the attacker tried to compromise these hosts (except the host *crow*[31]) by exploiting their vulnerabilities, apparently the attempts were unsuccessful. Since sensors typically cannot report on the success of the attacks, the fusion model justifiably uses them in correlation and reports them. However, absence of any further activity for these hosts resulted in low reported incident association strengths (27%) for them, suggesting less concern for their involvement in multi-staged attacks.

Using a similar Alert Reduction Rate metric as described earlier in section 4.3.1, Figure 4.8 shows the performance of the abstract alert correlation (AAC) approach in reducing the alert volume for the RealSecure-NCSU sensor report. The figure shows that abstract alert correlation further reduces the alert volume of prioritized alerts in terms of reporting only correlated alerts (i.e., alerts that are part of multi-staged attacks).

[31] The reason behind incorrectly correlating alerts for this host is explained earlier in this section.

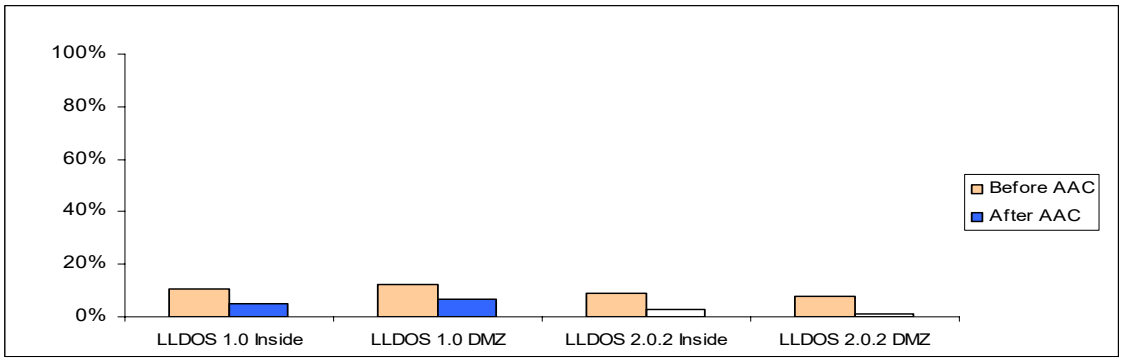


Figure 4.8 Alert Reduction with Abstract Alert Correlation (AAC) for RealSecure-NCSU Sensor Report

*R&A for RealSecure-MSU Sensor Report*

This report contained fewer alerts than the RealSecure-NCSU sensor report, although they were both generated with the same sensor monitoring the same attack traffic. There are two reasons for this discrepancy: the version of RealSecure software used and the choice of the sensor security policy applied. The NCSU researchers executed RealSecure Version 6.0 with the "Maximum Coverage" policy. At MSU, we executed RealSecure Version 7.0 with the "Attack Detection" policy. As mentioned before, the "Maximum Coverage" policy captures all activities (including security and connection events, filters and notifications), while the "Attack Detection" policy captures only definitive security events or attacks [21]. Such discrepancy helped us to better compare results from diversified sensor reports.

Table 4.4 shows the alert correlation results obtained from analyzing the RealSecure-MSU sensor report in terms of the correlation metrics described earlier.

Table 4.4 Correlation Performance for the RealSecure-MSU Sensor Report

| Dataset | SA | CR= CCA +MA | CA= CCA+ ICA | CCA | ICA | MA | TCR= CCA/CR | FCR= ICA/CA |
|---|---|---|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 93 | 19 | 16 | 16 | 0 | 3 | 84.21% | 0% |
| LLDOS 1.0 DMZ | 83 | 25 | 23 | 21 | 2 | 4 | 84.0% | 8.69% |
| LLDOS 2.0.2 Inside Zone | 41 | 10 | 8 | 8 | 0 | 2 | 80.0% | 0% |
| LLDOS 2.0.2 DMZ | 29 | 3 | 4 | 3 | 1 | 0 | 100.0% | 25.0% |

The column titled MA of Table 4.4 shows the number of alerts missed by the fusion model for each of the datasets. The following explains why the fusion model missed those alerts.

In the case of LLDOS 1.0 Inside Zone dataset, the fusion model missed the following alerts:

- Two *Stream_DoS* alerts launched from a spoofed IP address and targeted for a host outside the network. As explained for RealSecure-NCSU sensor report, the fusion model does not correlate alerts when the target host is outside of the protected network perimeter.
- One *Telnet_Auth_Failed* alert for the host *pascal:* 172.016.112.050. The fusion model did not correlate this alert because this alert activated a *Policy_Compliance_Suspicious* CEvent, which does not generate any incident that is part of a coordinated attack such as shown in Figure 3.7 of Chapter III.

For LLDOS 1.0 DMZ dataset, the fusion model missed the following alerts:

- One of two *IPSweep* alerts, which activated a *Disclosure_of_Host* (*DHS*) incident, for the designated network. The reason for missing this alert is that this particular alert followed all reported *Sadming_Ping* alerts, which activated a *Disclosure_of_Service* (*DSV*) incident. The time order between these incidents places the *DHS* incident after the *DSV* incident, thus implying that the attacker could not have used the knowledge gained by the *IPSweep* in the execution of the *ping* probes. Therefore, the fusion model did not correlate the *IPSweep* alert in the scenario.

- Three *Telnet_Auth_Failed* alerts for the hosts *locke:* 172.016.112.010, *plato:* 172.016.114.010, and *smith:* 172.016.114.020 for the same reason as in the case for LLDOS 1.0 Inside Zone dataset.

In the case of LLDOS 2.0.2 Inside Zone dataset, the fusion model missed two *Stream_DoS* alerts launched from a spoofed IP address and targeted for a host outside the network for similar reasons explained earlier.

The column titled ICA of Table 4.4 shows the number of alerts incorrectly correlated by the fusion model for each of the datasets. It should be noted that the fusion model did not incorrectly correlate any alerts for LLDOS 1.0 and LLDOS 2.0.2 Inside Zone datasets and thus achieved zero false causality rates in those cases. The following explains why the fusion model incorrectly correlated alerts for LLDOS 1.0 and LLDOS 2.0.2 DMZ datasets.

The number of incorrectly correlated alerts (ICA) is two in LLDOS 1.0 DMZ dataset and one in LLDOS 2.0.2 DMZ dataset. These alerts were of type *ICMP_Flood,* which were generated between the DMZ hosts *loud*: 172.016.114.001 and *marx:* 172.016.114.050. We found that large numbers of *ICMP_Redirect_Host* messages from host *loud* resulted in two *ICMP_Flood* alerts in LLDOS 1.0 DMZ dataset and one *ICMP_Flood* alert in LLDOS 2.0.2 DMZ dataset. Although clearly false positives in this

case, an *ICMP_Flood* alert can indicate an actual initiation of a *Denial of Service* attack. Therefore the fusion model activated a *System_Distress* incident and reported the hosts *loud* and *marx* under attack. While the reporting of these alerts is justifiable, we count them as false positives since these are not mentioned in the LLD documentation.

As mentioned before, because of the policy applied and the version used, the RealSecure-MSU sensor report differs from the RealSecure-NCSU sensor report. Abstract alert correlation by the fusion model helped us to understand these differences and their effects on overall incident assessment. For example, the overall incident situation for host *mill* in Figure 4.9 (analyzing the RealSecure-MSU sensor report), as compared with the one in Figure 4.7 (analyzing the RealSecure-NCSU sensor report), has some notable differences.


Figure 4.9 Incident Situation for Host *mill* for the RealSecure-MSU Sensor Report

Figure 4.9 shows a *Disclosure_of_Host* (DHS) incident activated by the fusion model in the case of all four datasets of the RealSecure-MSU sensor report, whereas, Figure 4.7 for RealSecure-NCSU sensor report shows low activation of such an incident by the fusion model for all of the datasets of the RealSecure-NCSU sensor report (the difference can also be seen in Table 4.5). This is because RealSecure Version 7.0 has attack signatures for the *IP_Sweep* attack while Version 6.0 does not. Therefore, with no evidence support, the incident was barely activated considering only pre-existing risk. Since the DHS incident does not present a critical impact on the system, the presence of such incidents in the case of RealSecure-MSU sensor report did not contribute greatly to the overall incident association strengths (IASs).

Table 4.5 Incident Situation for Host *mill* from analyzing the RealSecure-NCSU and the RealSecure-MSU Sensor Reports

| Dataset | Report | Incidents with Incident Strengths | | | | | IAS |
|---|---|---|---|---|---|---|---|
| | | DHS | DSV | SEC | SSZ | SDT | |
| LLDOS 1.0 Inside Zone | NCSU | 0.100 | 0.820 | 0.940 | 0.964 | 0.993 | 91.80 |
| | MSU | 1.00 | 1.00 | 1.00 | 0.600 | 0.920 | 70.80 |
| LLDOS 1.0 DMZ | NCSU | 0.100 | 0.820 | 0.940 | 0.964 | 0.193 | 52.10 |
| | MSU | 1.00 | 1.00 | 1.00 | 0.600 | 0.120 | 34.00 |
| LLDOS 2.0.2 Inside Zone | NCSU | 0.100 | 0.020 | 0.673 | 0.804 | 0.961 | 72.80 |
| | MSU | 1.00 | 0.200 | 0.733 | 0.440 | 0.888 | 54.20 |
| LLDOS 2.0.2 DMZ | NCSU | 0.100 | 0.020 | 0.673 | 0.804 | 0.161 | 34.40 |
| | MSU | 1.00 | 0.200 | 0.733 | 0.440 | 0.088 | 18.70 |

Table 4.5 compares the incidents reported by the fusion model for host *mill* from analyzing the RealSecure-NCSU and RealSecure-MSU sensor reports. The reason we show data from both of the reports is to point out the dependence of incident activation on evidence and associated risks. The pink cells in Table 4.5 indicate incidents activated with support of related evidence in the sensor reports and the yellow cells indicate incidents activated without any support of related evidence in the sensor reports. The values in yellow cells were computed considering only the pre-existing risks or the possibilities of the incidents occurring. The following are some interesting observations from Table 4.5. It should be noted that the incidents activated by the fusion model should be read from left to right in the table, as they occur chronologically in sequence.

- Change in cell color from yellow to pink in the table points out a notable rise in incident strength due to the addition of actual support evidence of the corresponding incident.
- Change in cell color from pink to yellow in the table points out a notable fall in incident strength due to a lack of actual support evidence of the corresponding incident.
- A string of consecutive pink cells indicates increasing incident strengths (unless they are at maximum, i.e., 1.0) as additional evidence of corresponding incidents continues to be accumulated with that of preceding incidents in the correlation chain.
- A string of consecutive yellow cells indicates decreasing incident strengths as absence of evidence of corresponding incidents continues to be accumulated with that of preceding incidents in the correlation chain.

For both of the sensor reports, Table 4.5 demonstrates how the strength of an incident or the extent to which an incident is activated depends on the support evidence for that incident and the risk of that incident occurring, which is computed from

activation of precedent incidents in the correlation chain. Table 4.5 also shows the incident association strengths (IAS) for host *mill* for each of the datasets in the RealSecure-NCSU and RealSecure-MSU sensor reports. It is reported to be highest (70.8%) for LLDOS 1.0 Inside Zone dataset of the RealSecure-NCSU sensor report (because of strong activation of all incidents in the correlation chain except *Disclosure_of_Host* incident) and lowest (18.7%) for LLDOS 2.0.2 DMZ dataset of the RealSecure-MSU sensor report (because of absences of critical incidents, such as *System_Seizure* and *System_Distress*). Interestingly, for the host *mill*, the IASs were reported lower for the RealSecure-MSU sensor report than for the RealSecure-NCSU sensor report (shown by the rightmost column of Table 4.5). The reason for the lower IASs is that since we at MSU applied the "Attack Detection" policy, notification type of alerts (like alerts for communication, for example, *telnet*, *remote shell*, *ftp* alerts), were not reported by the sensor. Hence without evidence of active communication, the *System_Seizure* incident was activated with lower strength. This contributed to low IASs reported for the hosts, as compared to the ones found for RealSecure-NCSU sensor report, where evidence of active communication found in the sensor report executed with the "Maximum Coverage" policy, contributed to raise the IASs for the hosts.

Table 4.6 Incident Association Assessment for RealSecure-MSU Sensor Report

| Dataset | Hosts | Incident Association Strength (IAS) |
|---|---|---|
| LLDOS 1.0 Inside Zone | **172.016.112.010** | **70.8%** |
| | **172.016.112.050** | **70.8%** |
| | **172.016.115.020** | **70.8%** |
| LLDOS 1.0 DMZ | **172.016.112.050** | **34.0%** |
| | **172.016.115.020** | **34.0%** |
| | 172.016.114.001 | 32.03% |
| | **172.016.114.010** | **34.0%** |
| | **172.016.114.020** | **34.0%** |
| | **172.016.114.030** | **34.0%** |
| | 172.016.114.050 | 32.03% |
| | 172.016.112.010 | 14.0% |
| LLDOS 2.0.2 Inside Zone | **172.016.115.020** | **54.90%** |
| | **172.016.112.050** | **48.70%** |
| LLDOS 2.0.2 DMZ | 172.016.114.001 | 32.03% |
| | 172.016.114.050 | 32.03% |
| | **172.016.115.020** | **18.67%** |

Table 4.6 shows a list of all reported hosts by the fusion model along with their IASs from analyzing the RealSecure-MSU sensor report. There are two hosts reported in this list (*loud:* 172.016.114.001 and *marx:* 172.016.114.050) that were not compromised (we have explained earlier why the fusion model incorrectly correlated alerts for these hosts). The IASs reported in the case of LLDOS 1.0 DMZ dataset show the same results (34%) for the hosts that are known (according to LLD documentation) to be actually compromised and the hosts that are known to be attacked but not compromised. This is because the sensor report contained the same set of alerts for all of these hosts and as the sensor can not report on the success of the attacks, the same IAS was reported for all of them. It should also be noted that in the case of LLDOS 1.0 DMZ dataset, the fusion model reported a lower IAS for one of the compromised hosts *locke:* 172.016.112.010, as

compared to the other compromised hosts. This is because RealSecure did not generate the same *Sadmind_Amslverify_Overflow* type of alert for this particular host as it did for the others for some unknown reason. Therefore, with no evidence of an *Access_Control_Violation* type of CEvent, considering only the pre-existing risk, the fusion model activated a *System_Environment_Corruption* incident with a low strength (0.33) for this host, as compared to the activation of the same incident for other hosts (1.0) in a similar situation. In fact, since there were no further subsequent alerts for this host, a low IAS (14%) was reported by the fusion model.

Using the Alert Reduction Rate metric, as described earlier, Figure 4.10 shows the performance of the abstract alert correlation (AAC) approach in reducing alert volume for the RealSecure-MSU sensor report. The figure shows that abstract alert correlation further reduces the alert volume of prioritized alerts in terms of reporting only correlated alerts (i.e., alerts that are part of multi-staged attacks).

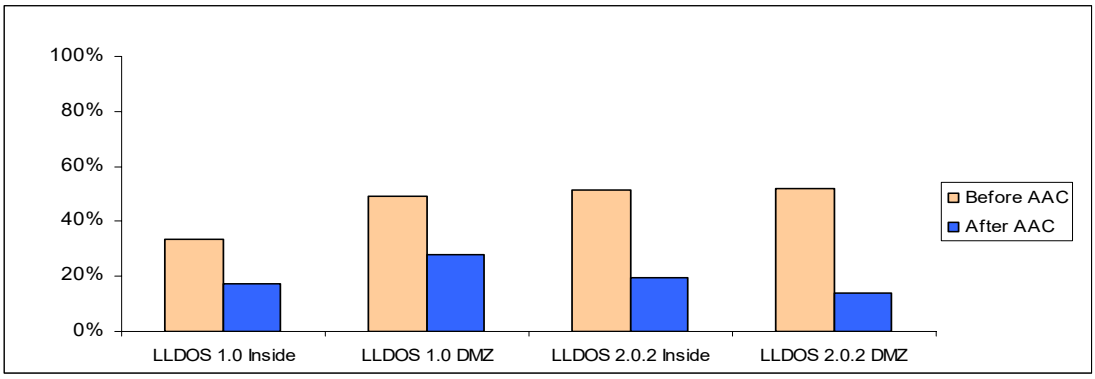


Figure 4.10 Alert Reduction with Abstract Alert Correlation (AAC) for RealSecure-MSU Sensor Report

*R&A for Snort-MSU Sensor Report*

Table 4.7 shows the abstract alert correlation results for the Snort-MSU sensor report in terms of the metrics described earlier.

Table 4.7 Correlation Performance for the Snort-MSU Sensor Report

| Dataset | SA | CR= CCA +MA | CA= CCA+ ICA | CCA | ICA | MA | TCR= CCA/CR | FCR= ICA/CA |
|---|---|---|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 1260 | 72 | 71 | 60 | 11 | 12 | 83.33% | 15.49% |
| LLDOS 1.0 DMZ | 3849 | 136 | 116 | 104 | 12 | 32 | 76.47% | 10.34% |
| LLDOS 2.0.2 Inside Zone | 879 | 16 | 15 | 15 | 0 | 1 | 93.75% | 0% |
| LLDOS 2.0.2 DMZ | 1479 | 6 | 6 | 6 | 0 | 0 | 100% | 0% |

As discussed before, according to the LLD documentation [35], telnet sessions were initiated by the attacker in preparation for installing DDoS tools on the compromised hosts. While counting causal relationships in Snort generated data, we consider three alerts for these telnet sessions (*Telnet_Access: TAC*, *Telnet_Login_Incorrect: TLI, and*: *INFO_TELNET_Bad_Login: TBL)*. The column titled MA of Table 4.7 shows the number of alerts missed by the fusion model for each of the datasets. The following explains why the fusion model missed those alerts.

In the case of LLDOS 1.0 Inside Zone dataset, the number of missed alerts (MA) was twelve, which are:

- Six alerts including one *ICMP_Echo_Reply (IER)* alert and two *telnet* alerts (*TLI* and *TBL*) for each of the two victim hosts *locke:* 172.016.112.010 and *pascal*: 172.016.112.050. In the fusion model, like the *Telnet_Authentication_Failure* (*TAF*) alert from RealSecure, *TLI* and *TBL* alerts from Snort activates a *Policy_Compliance_Suspicious* CEvent, which does not generate any incident that is part of a coordinated attack such as shown in Figure 3.7 of Chapter III. For this same reason, the alert fusion model also overlooked the *IER* alert. However, IER alerts are considered in counting causal relationships in data because these alerts denote that the hosts in question responded to *ping* requests by the attacker.

- Similarly five alerts including one *IER,* two *TLI* and two *TBL* alerts were missed for the host *mill:* 172.016.115.020 (because telnet sessions were initiated twice for this host).

- One *TAC* alert for the host *mill:* 172.016.115.020. This is because this particular alert activated a *System_Seizure* incident, which occurred after its successor *System_Distress* incident (as shown in the abstract incident model of Figure 3.7 of Chapter III). Because of the time order, the alerts were not correlated.

In the alert report for LLDOS 1.0 DMZ dataset, the attacker actively pursued six hosts (*locke:* 172.016.112.010, *pascal:* 172.016.112.050, *mill:* 172.016.115.020, *plato:* 172.016.114.010, *smith:* 172.016.114.020, and *solomon:* 172.016.114.030). The number of telnet sessions initiated for these hosts were: one each for *locke* and *pascal*, two for *mill*, three each for *plato*, *smith* and *solomon*. Since for each telnet session two alerts (one *TLI* and one *TBL*) were missed by the fusion model for reasons similar to those explained earlier, there were thirteen *TLI* alerts and thirteen *TBL* alerts missed in total. In addition to these twenty six alerts, six IER alerts were missed for the six hosts for reasons explained above. The presence of multiple telnet and ftp sessions in scenario one affected the causality rates considerably for LLDOS 1.0 Inside Zone and DMZ datasets.

For LLDOS 2.0.2 Inside Zone dataset, the fusion model missed one alert (*ICMP_Destination_Unreachable_Port_Unreachable),* because this alert activated *Policy_Compliance_Suspicious* CEvent, which does not generate any incident that is part of a coordinated attack scenario such as shown in Figure 3.7 of Chapter III.

The column titled ICA of Table 4.7 shows the number of alerts incorrectly correlated by the fusion model for each of the datasets. It should be noted that the fusion model did not incorrectly correlate any alerts for LLDOS 2.0.2 Inside Zone and DMZ datasets and thus zero false causality rates in those cases. The following explains why the fusion model incorrectly correlated alerts for LLDOS 1.0 Inside Zone and DMZ datasets.

The alert fusion model incorrectly correlated eleven alerts for LLDOS 1.0 Inside Zone dataset, which are:

- Six *ICMP_Ping (PNG)* alerts were correlated with two *Telnet_Access* (*TAC*) alerts for the host *falcon*: 172.016.112.194. The reason for this incorrect correlation is that for the same pair of hosts, the *TAC* alerts activated a *System_Seizure* incident that followed after a *Disclosure_of_Host* incident, which was activated due to the *PNG* alerts. Since two sequential incidents in a multi-staged attack (as shown in Figure 3.7 of Chapter III) occurred for this host, the corresponding alerts were correlated.

- One *WEB-CGI finger access (WFA)* alert and one *WEB-MISC /doc/ access (WMA)* alert incorrectly correlated with one *TAC* alert for the host *marx:* 172.016.114.050. The reason for this correlation is that *TAC* alerts activate a *System_Seizure* (SSZ) incident that followed after a *System_Environment_ Corruption* (SEC) incident – activated due to the *WFA* and *WMA* alerts. Therefore the corresponding alerts that contributed to these two sequential incidents in a multi-staged attack were correlated.

For LLDOS 1.0 DMZ dataset, twelve alerts (six *PNG* alerts activating a *Disclosure_of_Host* incident, followed by one *WFA* and one *WMA* alert activating a *System_Environment_ Corruption* incident, followed by four *TAC* alerts activating a

*System_Seizure* incident), were incorrectly correlated for the host *marx*. These alerts were all correlated because these contributed to generate sequential incidents in a multi-staged attack.

Figure 4.11 is a snapshot of the incident situation discovered for the compromised host *mill:* 172.016.115.020 from analyzing the Snort-MSU sensor report.

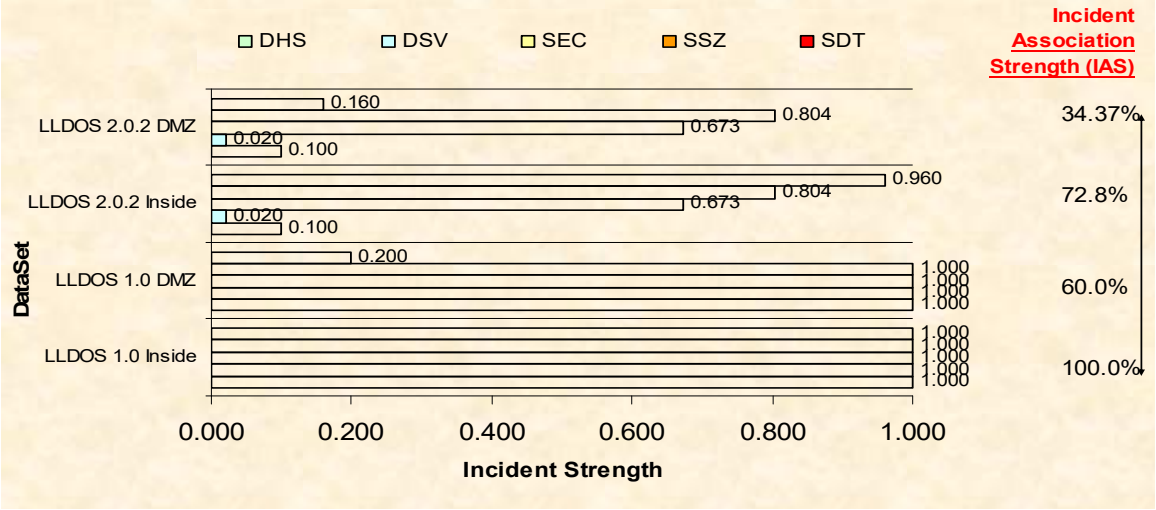


Figure 4.11 Incident Situation for Host *mill* from analyzing the Snort-MSU Sensor Report

As shown in Figure 4.11, the incident association strength (IAS) for *mill* is the maximum (100%) possible in the case of LLDOS 1.0 Inside Zone dataset (because of activation of all incidents in the multi-staged attack scenario) and the lowest (34.37%) in the case of LLDOS 2.0.2 DMZ dataset (because of the absence of evidence for the most detrimental incident, *System_Distress* (SDT) and two other incidents *Disclosure_of_Host* (DHS) and *Disclosure_of_Service* (DSV).

Table 4.8 shows a list of the hosts reported by the fusion model for all the datasets along with their IASs after analyzing the Snort-MSU sensor report. The non-shaded rows show two hosts that were not compromised according to the LLD documentation. These hosts *falcon*: 172.016.112.194 and *marx*: 172.016.114.050 were reported because of the alerts that were incorrectly correlated for them (explained earlier in the section). However, the low IAS reported for these hosts indicate that the incident situation for these hosts were not critical.

Table 4.8 Incident Association Assessment for the Snort-MSU Sensor Report

| Dataset | Hosts | Incident Association Strength (IAS) |
|---|---|---|
| LLDOS 1.0 Inside Zone | **172.016.112.010** | **100%** |
| | **172.016.112.050** | **100%** |
| | **172.016.115.020** | **100%** |
| | 172.016.112.194 | 15.44% |
| | 172.016.114.050 | 34.37% |
| LLDOS 1.0 DMZ | **172.016.112.010** | **60%** |
| | **172.016.112.050** | **60%** |
| | **172.016.115.020** | **60%** |
| | **172.016.114.010** | **60%** |
| | **172.016.114.020** | **60%** |
| | **172.016.114.030** | **60%** |
| | 172.016.114.050 | 34.37% |
| | 172.016.112.194 | 15.44% |
| LLDOS 2.0.2 Inside Zone | **172.016.112.050** | **72.8%** |
| | **172.016.115.020** | **72.8%** |
| LLDOS 2.0.2 DMZ | **172.016.115.020** | **34.37%** |

As described in Chapter III, with the abstract incident model for alert correlation the fusion model is able to compensate for missing alerts in sensor reports by taking into account the risk or the possibility of incidents occurring that are related to such alerts. In order to examine this capability of the abstract incident model, additional experiment with the Snort-MSU sensor report was performed for four different cases. Focusing on the host *mill:* 172.016.115.020, the experiment was conducted for LLDOS 1.0 Inside Zone dataset. In the first case (Case 1), there was no manipulation of the sensor report, where all original evidence or alerts were preserved for this particular host. In subsequent cases, we purposely excluded alerts from the sensor report in increments to show how the fusion model dealt with different combinations of missing alerts. The experimental cases were set up as follows:

- *Case 1*: Evidence of all incidents in the correlation chain were present in LLDOS 1.0 Inside Zone dataset of the Snort-MSU report for the host *mill*;
- *Case 2*: Evidence of the *SEC* incident was excluded from LLDOS 1.0 Inside Zone dataset for the host *mill*;
- *Case 3*: Evidence of the *SEC* and the *SSZ* incidents were excluded from LLDOS 1.0 Inside Zone dataset for the host *mill*; and
- *Case 4*: Evidence of the *SEC*, *SSZ* and *SDT* incidents were excluded from LLDOS 1.0 Inside Zone dataset for the host *mill*.

Figure 4.12 shows the incident situation discovered for host *mill* for these different cases.

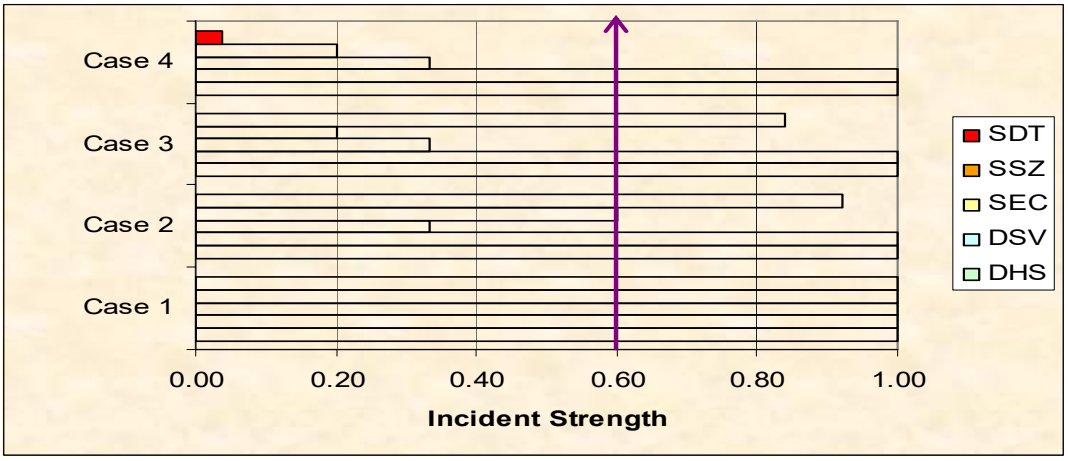


Figure 4.12 Incident Situation for Host *mill* with Missing Alerts from the Snort-MSU Sensor Report

Case 1 in Figure 4.12 shows that all incidents in the correlation chain were activated to their full extent because of support of all corresponding evidence (i.e., there were no related missing alerts). In case 2, even though there was no evidence of the *SEC* incident in the sensor report, the fusion model still activated this particular incident to some extent using the possibility of the incident occurring, by observing the status of its predecessor incidents in the correlation chain (Figure 3.7 in Chapter III). When more evidence of predecessor incidents in the correlation chain are missing, less activation of successor incidents occurs, even if there is evidence of support for them present in the sensor report. For example, the *SDT* incident had more missing alerts for predecessor incidents in case 3 than in case 2 (case 3 has missing alerts for the *SEC* and *SSZ* incidents and case 2 has missing alerts for the *SEC* incident). Therefore activation of the *SDT* incident in case 3 was less than in case 2. In case 4, since all evidence for incidents that followed the *DSV* incident were missing, Figure 4.12 shows that the strength of the successor incidents gradually decreased and eventually subsided. An effective threshold scheme can isolate

the incidents that are activated with support of evidence from those that are activated without evidence. For example, Figure 4.12 shows that any incidents that were activated with strength less than the threshold of 0.60, indicates that the incident had been activated considering only the risk, without support of any corresponding evidence in the sensor report. Figure 4.12 identifies such incidents below the threshold of 0.60 clearly.

Figure 4.13 shows the performance of the abstract alert correlation (AAC) approach in reducing alert volume for the Snort-MSU sensor report. The figure shows that abstract alert correlation further reduces the alert volume of prioritized alerts in terms of reporting only correlated alerts (i.e., alerts that are part of multi-staged attacks).

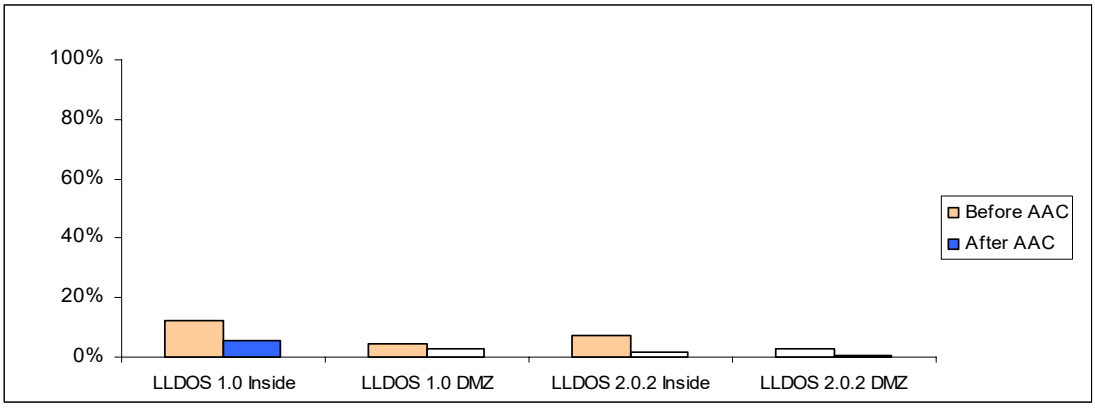


Figure 4.13 Alert Reduction with Abstract Alert Correlation (AAC) for Snort-MSU Sensor Report

*R&A for MultiSensor-MSU Report*

Table 4.9 denotes the alert correlation results for the multi-sensor report in terms of the metrics described earlier.

Table 4.9 Correlation Performance for the MultiSensor-MSU Report

| Dataset | SA | CR= CCA +MA | CA= CCA+ ICA | CCA | ICA | MA | TCR= CCA/CR | FCR= ICA/CA |
|---|---|---|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 1353 | 91 | 86 | 75 | 11 | 16 | 82.42% | 12.79% |
| LLDOS 1.0 DMZ | 3932 | 161 | 136 | 122 | 14 | 39 | 75.78% | 10.29% |
| LLDOS 2.0.2 Inside Zone | 920 | 26 | 22 | 22 | 0 | 4 | 84.61% | 0% |
| LLDOS 2.0.2 DMZ | 1508 | 9 | 10 | 9 | 1 | 0 | 100% | 10% |

It should be pointed out that the number of causal relations (CR) in the integrated multi-sensor reports (column 3 of Table 4.9) is the sum of the number of causal relations in the RealSecure-MSU sensor report (column 3 of Table 4.4) and the number of causal relations in the Snort-MSU sensor report (column 3 of Table 4.7). As the fusion model performed analysis on the multi-sensor data, the following results were expected:

- Alerts incorrectly correlated in intra-sensor fusion (analyzing the individual sensor reports) would also be incorrectly correlated in inter-sensor fusion (analyzing the multi-sensor report). This is because evidence present in individual sensor reports that leads to incorrect correlation, remain present when integrated. However, there should not be additional false positives. The only exception would be if incident situations are discovered when unrelated evidence found in individual sensor reports are linked together in the multi-sensor report to collectively discover a seemingly coordinated attack scenario.

- Alerts missed in intra-sensor fusion would also be missed with inter-sensor fusion. However, there should not be any additional false negatives unless there is related evidence found in individual sensor reports that cannot seemingly be linked together in the multi-sensor report to support a coordinated attack scenario.

As experiment was conducted on the multi-sensor report, the following was found:

- As expected, the fusion model incorrectly correlated the same set of alerts for inter-sensor fusion that were incorrectly correlated for intra-sensor fusion (in case of RealSecure-MSU and Snort-MSU sensor reports). That is, the number of incorrectly correlated alerts (ICA) in the integrated multi-sensor reports (column 6 of Table 4.9) are the sum of the number of incorrectly correlated alerts (ICA) in the RealSecure-MSU sensor report (column 6 of Table 4.4) and the number of incorrectly correlated alerts (ICA) in the Snort-MSU sensor report (column 6 of Table 4.7).

- Unexpectedly, there were additional missed alerts (shown in the last column of Table 4.10) beyond those missed for RealSecure-MSU and Snort-MSU sensor reports. That is, the number of missed alerts (MA) in the integrated multi-sensor reports (column 7 of Table 4.9) is greater than the sum of the number of missed alerts (MA) in the RealSecure-MSU sensor report (column 7 of Table 4.4) and the number of missed alerts (MA) in the Snort-MSU sensor report (column 7 of Table 4.7). Table 4.10 shows the additional missed alerts.

Table 4.10 Comparison of Correlated Alerts found for RealSecure-MSU Report, Snort-MSU Report and MultiSensor-MSU Report

| Dataset | Causal Alerts (CA) for RealSecure-MSU | CA for Snort-MSU | Expected CA for MultiSensor MSU= CA for RealSecure-MSU + CA for Snort-MSU | Actual CA Multi Sensor MSU | Additional Missed Alerts= Expected CA - Actual CA |
|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 16 | 71 | 87 | 86 | 1 |
| LLDOS 1.0 DMZ | 23 | 116 | 139 | 136 | 3 |
| LLDOS 2.0.2 Inside Zone | 8 | 15 | 23 | 22 | 1 |
| LLDOS 2.0.2 DMZ | 4 | 6 | 10 | 10 | 0 |

It should be noted that the discrepancy in the number of causal or correlated alerts for the RealSecure-MSU sensor report (column 2 of Table 4.10) and the Snort-MSU sensor report (column 3 of Table 4.10) is due to the fact that the sensors were executed with different security policies (as explained in section 4.2 earlier). In the case of LLDOS 1.0 Inside Zone dataset, the one additional missed alert was a *Sadmind_Buffer_Overflow* alert, reported by RealSecure and generated for the host *locke:* 172.016.112.010. This alert activated a *System_Environment_Corruption* incident for this host. However, since this incident occurred after its successor *System_Seizure* incident (activated due to alerts reported by Snort) and not before it, the fusion model found the incidents unrelated. In the cases of LLDOS 1.0 DMZ and LLDOS 2.0.2 Inside Zone datasets, the following alerts were missed by the fusion model for the same reason:

- For LLDOS 1.0 DMZ dataset, three *Sadmind_Buffer_Overflow* alerts reported by RealSecure for the hosts *plato:* 172.016.114.010, *smith:* 172.016.114.020 and *solomon:* 172.016.114.030;
- For LLDOS 2.0.2 Inside Zone dataset, one *Sadmind_Buffer_Overflow* alert reported by RealSecure for the host *pascal:* 172.016.112.050.

For comparison purposes, Table 4.11 shows a list of all hosts reported by the fusion model after analyzing the RealSecure-MSU Report (Experiment 4.3.2B), Snort-MSU Report (Experiment. 4.3.2C), and MultiSensor Report (Experiment. 4.3.2D). The listings in <BOLD> indicate the actual attacked hosts in the LLD experiments. The abbreviations used in this table are:

- R: RealSecure-MSU Sensor Report;
- S: Snort-MSU Sensor Report;
- M: MultiSensor-MSU Report.

Table 4.11 Comparison of Incident Situation Discovered after Analyzing RealSecure-MSU Report, Snort-MSU Report, and MultiSensor-MSU Report

| Dataset | Host | DHS Incident Activated for | DSV Incident Activated for | SEC Incident Activated for | SSZ Incident Activated for | SDT Incident Activated for | IAS found for R | IAS found for S | IAS found for M |
|---|---|---|---|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | **172.016.112.010** | R/S/M | R/S/M | R/S/M | S/M | R/S/M | 70.8 | 100.0 | 100.0 |
| | **172.016.112.050** | R/S/M | R/S/M | R/S/M | S/M | R/S/M | 70.8 | 100.0 | 100.0 |
| | **172.016.115.020** | R/S/M | R/S/M | R/S/M | S/M | R/S/M | 70.8 | 100.0 | 100.0 |
| | 172.016.112.194 | S/M | | | S/M | | | 15.44 | 15.44 |
| | 172.016.114.050 | | | S/M | S/M | | | 34.37 | 34.37 |
| LLDOS 1.0 DMZ | **172.016.112.010** | R/S/M | R/S/M | R/S/M | S/M | | 14.0 | 60.0 | 60.0 |
| | **172.016.112.050** | R/S/M | R/S/M | R/S/M | S/M | | 34.0 | 60.0 | 60.0 |
| | **172.016.115.020** | R/S/M | R/S/M | R/S/M | S/M | | 34.0 | 60.0 | 60.0 |
| | **172.016.114.010** | R/S/M | R/S/M | R/S/M | S/M | | 34.0 | 60.0 | 60.0 |
| | **172.016.114.020** | R/S/M | R/S/M | R/S/M | S/M | | 34.0 | 60.0 | 60.0 |
| | **172.016.114.030** | R/S/M | R/S/M | R/S/M | S/M | | 34.0 | 60.0 | 60.0 |
| | 172.016.112.194 | S/M | | | S/M | | | 15.44 | 15.44 |
| | 172.016.114.001 | | | | | R/M | 32.03 | | 32.03 |
| | 172.016.114.050 | | | S/M | S/M | R/M | 32.03 | 34.37 | 72.8 |
| LLDOS 2.0.2 Inside Zone | **172.016.112.050** | | | R/S/M | S/M | R/S/M | 48.7 | 72.8 | 72.8 |
| | **172.016.115.020** | R/M | | R/S/M | S/M | R/S/M | 54.19 | 72.8 | 79.23 |
| LLDOS 2.0.2 DMZ | 172.016.114.001 | | | | | R/M | 32.03 | | 32.03 |
| | 172.016.114.050 | | | | | R/M | 32.03 | | 32.03 |
| | **172.016.115.020** | R/M | | R/S/M | S/M | | 18.67 | 34.37 | 40.51 |

Table 4.11 shows some interesting cases where incidents were linked together with inter-sensor fusion:

- In the case of LLDOS 1.0 Inside Zone dataset, for each of the three compromised hosts, the fusion model was able to correlate *DHS*, *DSV*, *SEC* and *SDT* incidents from evidence reported by both RealSecure and Snort with a *SSZ* incident inferred only from evidence reported by Snort. Since the Snort-MSU report also had evidence of all related incidents, the IAS reported for the MultiSensor-MSU report is the same as that for the Snort-MSU sensor report.

- In the case of LLDOS 1.0 DMZ dataset, for one of the host *marx:* 172.016.114.050, the fusion model correlated *SEC* and *SSZ* incidents from evidence reported only by Snort with a *SDT* incident from evidence reported only by RealSecure. Figure 4.14 shows the incident situation for this host. Although these alerts came from different sensors (i.e., RealSecure and Snort), the fusion model linked them together because the alerts corresponded to sequential incidents in a multi-staged attack such as shown in Figure 3.7 of Chapter III. This correlation is justifiable because we are interested in the ultimate impact of security incidents on a target and it is feasible for a target to be attacked from multiple sources. Therefore, such correlation is needed for comprehensive security analysis. In this case (Figure 4.15), as a result of this correlation, the IAS reported was higher (72.8%) as compared to what were found analyzing the RealSecure report (32.03%) and the Snort report (34.37%).

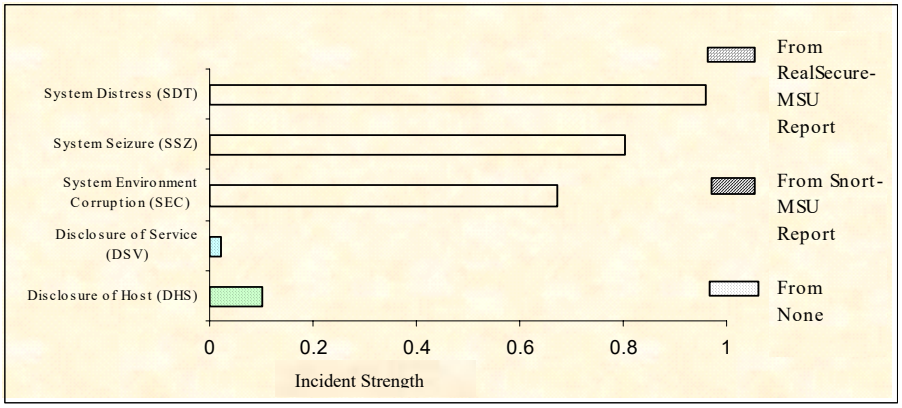


Figure 4.14 Incident Situation for Host *marx:* 172.015.114.050 analyzing the MultiSensor-MSU Report

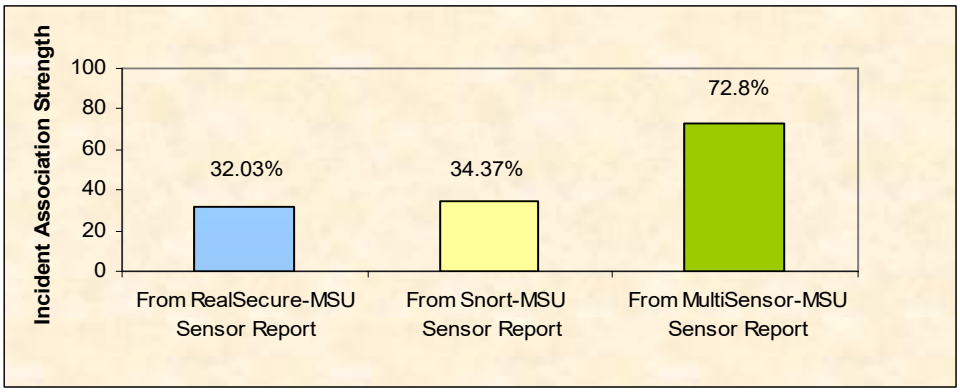


Figure 4.15 Comparison of IAS reported for Host *marx:* 172.015.114.050 for RealSecure-MSU, Snort-MSU and MultiSensor-MSU reports

- In the case of LLDOS 1.0 DMZ dataset, for the hosts under attack, the fusion model was able to correlate *DHS*, *DSV* and *SEC* incidents inferred from evidence reported by RealSecure and Snort with the *SSZ* incident inferred from evidence reported only by Snort. Since the Snort-MSU report also had evidence of all related incidents, the IAS reported for the MultiSensor-MSU report is the same as that for the Snort-MSU sensor report.

- For the host *falcon:* 172.016.112.194, in the case of LLDOS 1.0 Inside Zone dataset, since the incidents activated because of evidence supported by Snort, the IAS reported remains the same as that for the Snort-MSU report. The same happens for the host *loud*: 172.016.114.001, in case of the LLDOS 1.0 and LLDOS 2.0.2 DMZ datasets, with sensor evidence provided by RealSecure.

- In the case of LLDOS 1.0 Inside Zone and DMZ datasets, the incident situation for the host *mill:* 172.016.115.020, is found to be the same as that in the Snort-MSU sensor report. This is because the multi-sensor report for LLDOS 1.0 Inside Zone and DMZ datasets did not provide any additional evidence other than what already existed in the Snort-MSU sensor report. However, the IAS is reported higher in the case of LLDOS 2.0.2 Inside Zone and DMZ datasets of the multi-sensor report than for the respective datasets of the individual sensor reports. The following explains why.

Table 4.12 Comparison of Incident Situation for Host *mill* analyzing LLDOS 2.0.2 Inside Zone Dataset of RealSecure-MSU Report, Snort-MSU Report, and MultiSensor-MSU Report

| Dataset | Analysis based on Sensor Report | DHS | DSV | SEC | SSZ | SDT | IAS |
|---|---|---|---|---|---|---|---|
| LLDOS 2.0.2 Inside Zone | RealSecure | 1.0 | 0.2 | 0.733 | 0.44 | 0.888 | 54.19% |
| | Snort | 0.1 | 0.02 | 0.673 | 0.8 | 0.961 | 72.80% |
| | Multi-Sensor | 1.0 | 0.2 | 0.733 | 0.84 | 0.968 | 79.23% |

Table 4.12 shows the incident situation for host *mill* in the case of LLDOS 2.0.2 Inside Zone dataset of RealSecure-MSU, Snort-MSU and MultiSensor-MSU reports. The yellow cells denote incidents activated based only on pre-existing risks and without any evidence from sensor reports. For example, none of the sensors reported any evidence of a *DSV* incident for this host. The blue cells in the last row indicate that in these cases, the fusion model complemented failure of one sensor in reporting an alert for a certain type of incident with another sensor reporting an alert for similar type of incident. The green cells in last row indicate both sensors reporting evidence of the same incident. For the multi-sensor report (last row in Table 4.12), the reported strengths of incidents *DHS*, *DSV* and *SEC*, are the same as the maximum of the corresponding incidents' strengths reported for the individual sensor reports. This is because of the same evidence support and existing risk conditions. However, for the multi-sensor report, the reported strengths of incidents *SSZ* and *SDT* are higher than the corresponding incidents' strengths reported for the individual sensor reports. In these cases, although the supporting evidence was the same, the existing risks were higher in the case of the multi-sensor report (since the predecessor cases of the incidents activated to a higher degree). Therefore, the successor

incidents (*SSZ* and *SDT*) were also activated to a higher degree and as a result, the overall IAS was reported higher (79.23%) than those found from analyzing the individual sensor reports. Figure 4.16 illustrates the incident situation for host *mill* discovered from analyzing the RealSecure-MSU, the Snort-MSU and the MultiSensor-MSU Report for LLDOS 2.0.2 Inside Zone dataset.

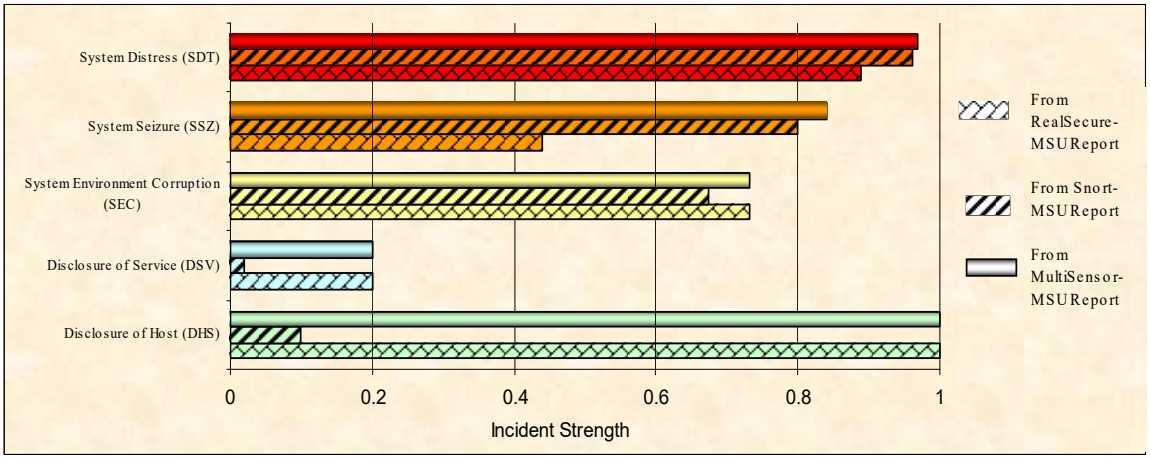


Figure 4.16 Comparison of Incident Situation for Host *mill* from analyzing the RealSecure-MSU, the Snort-MSU and the MultiSensor-MSU Report for LLDOS 1.0 Inside Zone Dataset

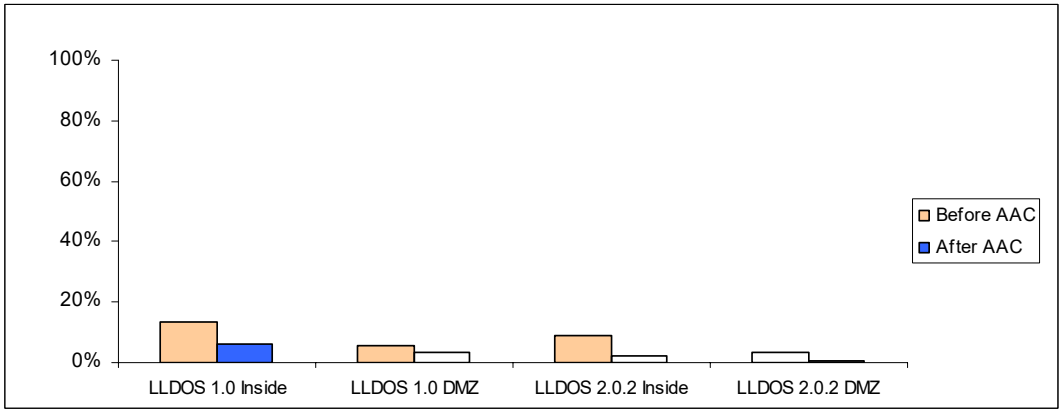


Figure 4.17 Alert Reduction with Abstract Alert Correlation (AAC) for MultiSensor-MSU Report

Figure 4.17 shows the performance of the abstract alert correlation (AAC) approach in reducing the alert volume for the MultiSensor-MSU report. The figure shows that abstract alert correlation further reduces the alert volume of prioritized alerts in terms of reporting only correlated alerts (i.e., alerts that are part of multi-staged attacks).

*Summary for Alert Correlation Experiment*

In the experiment conducted for abstract alert correlation, we found that the fusion model was able to correlate alerts that were generated as part of a coordinated attack scenario. While correlating alerts or finding causal relationship between alerts, it also reported on security incidents that had occurred for the hosts involved in the attacks. The extent of incident activation depended on evidence supporting the incident and the risk or the possibility of the incident occurring. That is, a high incident value indicated the presence of both evidence and risk for the incident and a low incident value indicated the absence of either the evidence or the risk. For each host reported under attack, an overall degree of concern for incident association was also reported. For example, a high incident association strength reported for a host indicated that one or more highly critical security incidents had occurred for the host and a low incident association strength reported for a host indicated that one or more less critical security incidents had occurred for the host. Thus incident association strengths and incident strengths provided the security administrator with an insight into the extent of concern for hosts involved in multi-staged attacks carried out by attacker.

### *4.3.3 Alert Clustering Experiment*

*Objective*

The purpose of this experiment was to evaluate the capability of the alert fusion model to cluster alerts with the same or similar features in order to determine a resource's involvement in common attack patterns. The experiment was conducted on four sensor alert reports - RealSecure-NCSU, RealSecure-MSU, Snort-MSU and MultiSensor-MSU individually.

*Evaluation*

The performance of any clustering approach can primarily be evaluated two ways [52] -using external quality measures, i.e., external knowledge about data (for example, class labels) or using internal quality measures (no outside knowledge). Since the fusion model analyses the sensor reports without apriori knowledge of any associations in data, external quality metrics such as entropy, purity or f-measure, which require accurate group/cluster labeling information, cannot be used in this case. Therefore, we used internal quality metrics to evaluate the performance of the multi-level alert clustering approach in terms of its coverage and quality. Measuring coverage is imperative to show that the multi-level clustering approach can extend a cluster's perimeter more than traditional exact clustering - in terms of accommodating additional meaningful data. Cluster quality, which is indicative of the cohesiveness of alerts in clusters, is important to evaluate to show that the multi-level clustering approach is able to produce clusters of alerts with features that are "somehow" alike, if not exactly.

The clustering metrics used in this dissertation are defined as follows:

- Cluster Coverage (CC): This metric measures the span of the low-level sensor alerts clustered. It is computed as the ratio of all clustered alerts to all prioritized[32] alerts. Therefore, cluster coverage is:

  $CC = \frac{n_c}{n_p}$, where $n_c$ is the number of the alerts clustered and $n_p$ is the number of the prioritized alerts before clustering.

  Since multi-level clustering groups additional meaningful alerts into more clusters than traditional exact clustering does, cluster coverage for multi-level clustering is likely to be higher than for traditional exact clustering.

- Intra-Cluster Similarity (ICS): Internal similarity of a cluster is an indication of the quality of the cluster and can be measured in two ways [52]. Intra-cluster similarity measures the cohesiveness of data in the cluster and inter-cluster similarity measures the distance between the clusters themselves. In our case, measuring the similarity of data in the clusters makes more sense because the objective of multi-level clustering is to find clusters with interrelated alerts. Therefore, only the intra-cluster similarity metric is used. This is a measure of the similarity or closeness of alerts in a cluster. We use the strength of the cluster, as defined in section 3.2.2.1 of Chapter III, as this intra-cluster similarity (ICS) measure. It is computed as the average of the candidacy scores of all contributing alerts in the cluster. Therefore, for a particular host, intra-cluster similarity for a cluster $c$ is:

  $ICS_c = \frac{\sum_{l} cs_{al} * n_{al}}{n_{ac}}$, where $n_{ac}$ is the total number of alerts with different similarity in the cluster $c$, $n_{al}$ is the number of alerts clustered at a particular level of similarity $l$ and $cs_{al}$ is the candidacy score for alerts in that particular level.

  For multi-level alert clusters, which incorporate similar alerts (alerts with similar features), intra-cluster similarity is likely to be lower than that found for traditional exact clusters, which only incorporate the same alerts (i.e., alerts with the same features).

[32] Alerts after filtering out low priority alerts with alert prioritization (section 3.2.1 of Chapter III).

– Cluster Overall Similarity (COS): As discussed above, while the cluster coverage of a multi-level cluster is expected to be higher than a traditional exact cluster, the intra-cluster similarity is expected to be lower. Therefore, another metric is used to highlight the tradeoff between the cluster coverage and intra-cluster similarity metrics. Cluster overall similarity denotes the weighted similarity of the internal clusters. Adapted from [52], for a particular host, cluster overall similarity for all clusters found for that host is computed as:

$COS = \sum_{c} CC_c * ICS_c$ where $CC_c$ is the cluster coverage of each cluster $c$ found for the host and $ICS_c$ is the intra-cluster similarity of the cluster $c$.

Since multi-level clustering incorporates additional similar alerts in more clusters than does traditional exact clustering, cluster overall similarity for multi-level clustering is likely to be higher than for traditional exact clustering.

In this dissertation, clustering results are reported in terms of cluster coverage per sensor report, cluster coverage per host and cluster overall similarity[33] per host in the sensor report. In addition to these metrics, since alert clustering also aids in alert reduction by reporting alert clusters, results are also shown for alert reduction using a similar alert reduction metric as discussed earlier for alert prioritization experiment.

*Results and Analysis (R&A)*

The following are the results and analysis of the alert clustering experiment. It should be noted that multi-level clustering is conducted on alerts filtered with alert prioritization.

[33] It should be noted that measuring Cluster Overall Similarity (COS) requires computation of the Intra-Cluster Similarity (ISC) measure.

*R&A for RealSecure-NCSU Sensor Report*

In the case of LLDOS 1.0 Inside Zone dataset, with traditional exact clustering (i.e., without multi-level clustering (MLC)), the alert fusion model was unable to cluster any of the 97 alerts in the sensor report. However with MLC, the fusion model was able to cluster 64 alerts for 11 of the hosts. In the case of LLDOS 1.0 DMZ dataset, without MLC, the alert fusion model only clustered 6 alerts for 1 particular host out of 108 alerts. With MLC, the number of clustered alerts increases to 60 for 11 additional hosts. In the case of LLDOS 2.0.2 Inside Zone dataset, without MLC, none of the 43 sensor alerts were clustered but with MLC, 34 alerts were clustered for 10 hosts. In the case of LLDOS 2.0.2 DMZ dataset, without MLC, the fusion model did not find any clusters from the 34 alerts in the report but with MLC, it clustered 25 alerts for 9 hosts. Figure 4.18 shows the cluster coverage results with and without MLC for all four datasets in the RealSecure-NCSU sensor report.

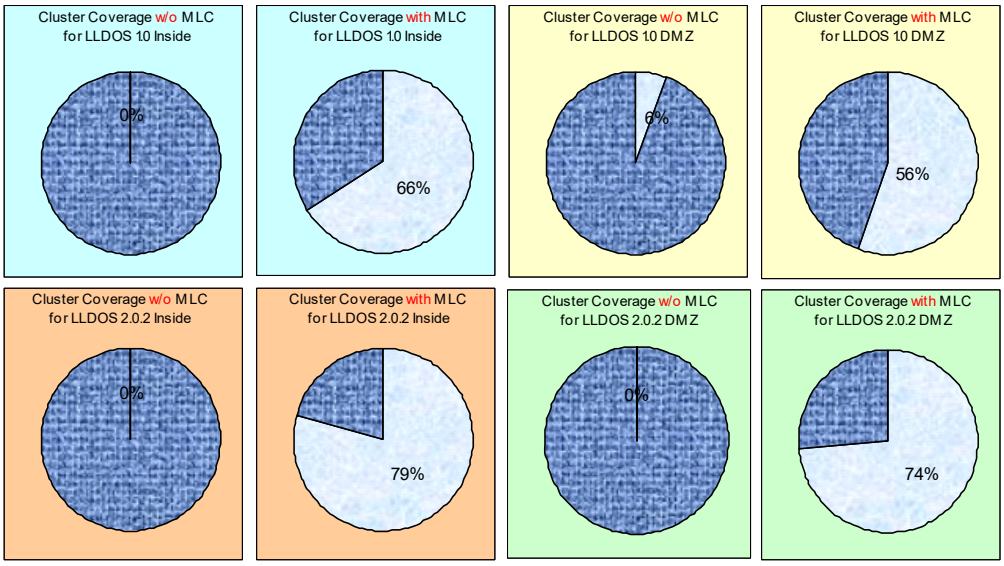


Figure 4.18 Comparison of Cluster Coverage with and without MLC for all the Datasets in the RealSecure-NCSU Sensor Report

The charts in Figure 4.18 show that with MLC, the alert fusion model was able to cluster more alerts than it did without MLC. Besides clustering of related alerts, the alert fusion model also reports cluster association strengths (CAS) for the hosts for which alert clusters are found. CAS is indicative of a host's involvement in common attack patterns. Table 4.13 lists the hosts, for which alert clusters were found analyzing all four datasets in the RealSecure-NCSU sensor report, with their cluster association strengths.

Table 4.13 Cluster Association Assessment for RealSecure-NCSU Sensor Report

| Dataset | Host | Cluster Association Strength (CAS) | Dataset | Host | Cluster Association Strength (CAS) |
|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | **172.016.115.020** | **75.94** | LLDOS 1.0 DMZ | **172.016.115.020** | **75.94** |
| | 172.016.112.100 | 73.64 | | 172.016.112.100 | 74.62 |
| | **172.016.112.050** | **72.88** | | 172.016.113.148 | 72.46 |
| | 172.016.113.148 | 72.06 | | **172.016.114.010** | **69.00** |
| | **172.016.112.010** | **65.51** | | **172.016.112.010** | **65.51** |
| | 172.016.112.149 | 58.60 | | **172.016.112.050** | **65.51** |
| | 172.016.113.204 | 58.60 | | **172.016.114.020** | **65.51** |
| | 172.016.113.169 | 58.60 | | **172.016.114.030** | **65.51** |
| | 172.016.112.194 | 58.60 | | 172.016.112.149 | 58.60 |
| | 172.016.113.168 | 58.60 | | 172.016.113.084 | 58.60 |
| | 172.016.113.084 | 58.60 | | 172.016.113.168 | 58.60 |
| | | | | 172.016.113.169 | 58.60 |
| LLDOS 2.0.2 Inside Zone | **172.016.112.050** | **76.62** | LLDOS 2.0.2 DMZ | **172.016.115.020** | **75.94** |
| | 172.016.112.100 | 75.94 | | 172.016.112.100 | 65.51 |
| | **172.016.115.020** | **75.94** | | 172.016.112.194 | 65.51 |
| | 172.016.112.194 | 65.51 | | 172.016.112.207 | 65.51 |
| | 172.016.112.207 | 65.51 | | 172.016.113.084 | 65.51 |
| | 172.016.113.084 | 65.51 | | 172.016.113.105 | 65.51 |
| | 172.016.113.105 | 65.51 | | 172.016.113.148 | 65.51 |
| | 172.016.113.148 | 65.51 | | 172.016.113.169 | 65.51 |
| | 172.016.113.169 | 65.51 | | 172.016.113.204 | 65.51 |
| | 172.016.113.204 | 65.51 | | | |

The shaded rows show the hosts for which highest CAS were reported in each dataset. The listings in <BOLD> show the hosts that were actively pursued by the attacker in the LLD experiments. It should be noted that a higher CAS for a particular host indicates that alerts were found to belong to multiple similar (or same) clusters - indicating a high concern for the host's involvement in common attack patterns. It should also be noted that, as described in section 3.2.2.1 of Chapter III, CAS is a weighted average of the alert clusters whose strengths are determined by the effort spent in generalizing the alerts to determine their fitness into the clusters. Therefore, CAS depends more on the number of clusters and the "quality" of such clusters than on the number of the alerts in the clusters.

Interesting observations made from the results shown in Table 4.13 are presented below:

For each of the datasets, CAS reported for the host *mill*: 172.112.115.020 is 75.94% (for LLDOS 1.0 Inside Zone and DMZ datasets, and LLDOS 2.0.2 DMZ dataset, it is the highest reported and for LLDOS 2.0.2 Inside Zone dataset, next to the highest). As mentioned earlier, this host is one of the victim hosts in the LLD attack experiments, which was actively pursued and compromised in the attack and also played a role in compromising other hosts in the network. After analyzing the sensor report, the alert fusion model discovered the following two types of clusters for this host for each of the datasets:

- *SameSource_SameAttack_RecentTime;* and
- *SameSource_SimilarAttack_RecentTime.*

It should be noted that both of these clusters are generalized and could only be found by MLC. The second cluster is more generalized than the first one. The first one grouped alerts generalizing only one feature attribute (i.e., time of the attack) and the second one grouped alerts by generalizing two feature attributes (nature and time of the attack), and therefore, more effort had to be spent on finding the second cluster than the first one. Hence the strength (or intra-cluster similarity) of the second cluster (0.688) was computed to be less than the first (0.801) one.

A close examination of the clusters generated for the host *mill* reveals that, for one of the datasets (i.e., LLDOS 1.0 DMZ dataset) without MLC the alert fusion model did not cluster any of the 11 prioritized alerts, while with MLC, it clustered 10 of them. The alerts found in the clusters were:

- Two *SameSource_SameAttack_RecentTime* type clusters. One consisting of six *Sadmind_Amslverify_Overflow* alerts, which did not occur at the same time (i.e., within seconds) but at similar time (i.e., within minutes). The other cluster consists of two *Remote_Shell* alerts occurring within minutes of each other.
- One *SameSource_SimilarAttack_RecentTime* type cluster including one *TelnetXdisplay* alert and one *TelnetEnvAll* alert. The alerts were clustered in this way because they were both *Active_Communication* type and were generated within close time proximity.

Charts 1 and 2 in Figure 4.19 compare the cluster coverage and cluster overall similarity for host *mill*: 172.016.115.020 with and without MLC for all four datasets in the RealSecure-NCSU sensor report. Since no alerts were clustered without MLC, both cluster coverage and cluster overall similarity are zero for that case. Therefore, Figure 4.19 shows notable improvement with MLC.

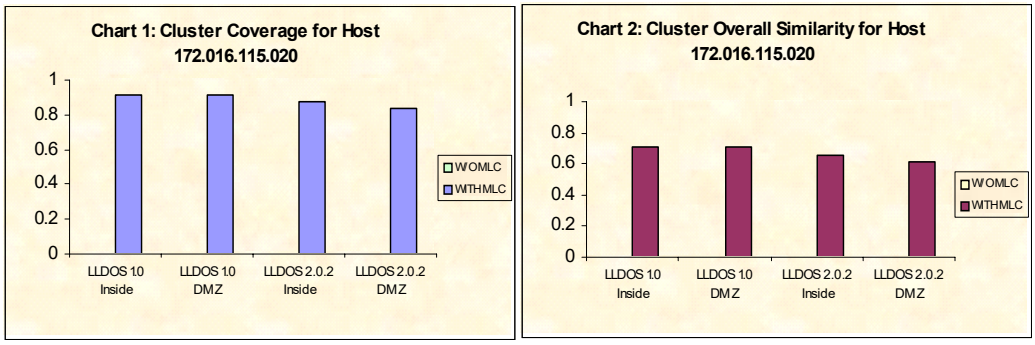


Figure 4.19 Cluster Coverage and Cluster Overall Similarity for Host *mill* analyzing the RealSecure-NCSU Sensor Report

In Table 4.13, we find that the host with the second highest CAS reported for all the datasets is an inside host *hume*: 172.016.112.100. Although this host did not play any role in the multi-staged attack, except the fact that it was probed initially [34], RealSecure generated multiple *FTP_Syst* alerts for this host, which came from different hosts, both inside and outside of the network. This type of alert is considered noteworthy by the alert fusion model because it indicates probing of service and therefore these alerts were included for cluster analysis. In the case of LLDOS 1.0 Inside Zone dataset, the fusion model analyzed 21 prioritized alerts for this host. Among them, without MLC, the fusion model did not cluster any alerts. However, with MLC the alert fusion model was able to cluster 18 of the alerts that matched "somewhat" in alert features. The different types of clusters found were:

– Six *SameSource_SameAttack_RecentTime* type clusters containing thirteen *FTP_Syst* alerts from the same set of sources occurring at similar times (i.e., within minutes, within hours, or within period).

– Three *SimilarSource_SameAttack_RecentTime* type clusters. One consisted of two *FTP_Syst* alerts originating from the same network address range (i.e., 172.016.113.xxx) and occurring within close proximity of previously clustered alerts. Another one included an *FTP_Syst* alert originating from the same class C IP address range (i.e., 196.xxx.xxx.xxx) and occurring within hours of previously clustered alerts. The third one consisted of two *FTP_Syst* alerts that originated from the same class B IP address range (i.e., 135.xxx.xxx.xxx) and occurred within hours of each other.

Table 4.14 shows the CAS computed for the host *hume* for all datasets in the RealSecure-NCSU sensor report along with the clusters generated and their strengths.

Table 4.14 Cluster Association of Host *hume* for the RealSecure-NCSU Sensor Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_RecentTime* | 0.759 | 73.64 |
| | *SimilarSource_SameAttack_RecentTime* | 0.567 | |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_RecentTime* | 0.763 | 74.62 |
| | *SimilarSource_SameAttack_RecentTime* | 0.632 | |
| LLDOS 2.0.2 Inside Zone | *SameSource_SameAttack_RecentTime* | 0.801 | 75.94 |
| | *SimilarSource_SameAttack_RecentTime* | 0.688 | |
| LLDOS 2.0.2 DMZ | *SameSource_SameAttack_RecentTime* | 0.801 | 65.50 |

Table 4.14 shows that although the same types of clusters were generated for each of the datasets, there are differences in the cluster strengths. These differences result from the fact that alerts in the clusters had to be generalized at different levels of abstraction to find feature similarity between them. For example, the strength (0.801) of the clusters of type *SameSource_SameAttack_RecentTime* found for LLDOS 2.0.2 Inside Zone and DMZ datasets (rows 5 and 7 of Table 4.14) is the highest among the strengths of the same type of clusters found for LLDOS 1.0 Inside Zone and DMZ datasets (row 1 and 3 of Table 4.14). This is because for LLDOS 2.0.2 Inside Zone and DMZ datasets, the clusters

contained alerts that had a time feature similarity at abstraction level 3 (Figure 3.3 in Chapter III). The same type clusters for LLDOS 1.0 DMZ dataset (strength 0.763) contained alerts that had a time feature similarity at abstraction levels 2 and 3. For LLDOS 1.0 Inside Zone dataset, the cluster (strength 0.759) contained alerts that had a time feature similarity at abstraction levels 1, 2 and 3. These results support the notion of cluster strength and indicate that cohesiveness (intra-similarity) of alerts in a cluster decreases as less similar alerts join the cluster. Charts 1 and 2 in Figure 4.20 compare the cluster coverage and cluster overall similarity for host *hume* with and without MLC for all four datasets in the RealSecure-NCSU sensor report. Since no alerts were clustered without MLC, Figure 4.20 shows clear improvement for both cluster coverage and cluster overall similarity with MLC.

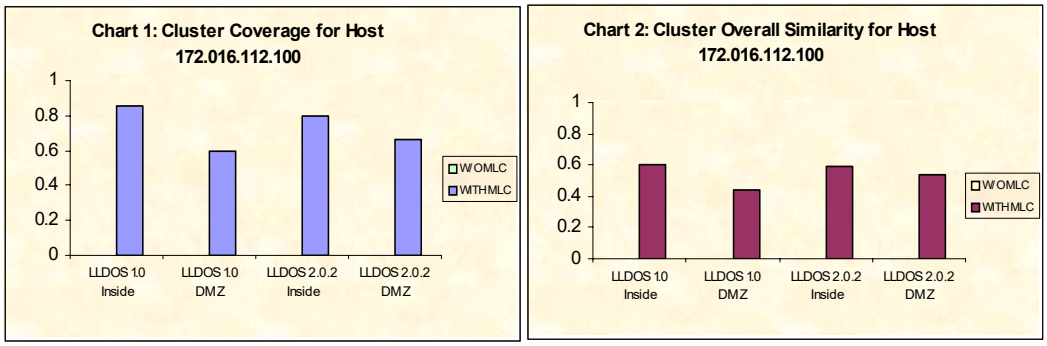


Figure 4.20 Cluster Coverage and Cluster Overall Similarity for Host *hume* analyzing the RealSecure-NCSU Sensor Report

For all the datasets in the RealSecure-NCSU sensor report, without MLC, the alert fusion model clustered alerts for only LLDOS 1.0 DMZ dataset and for only one host, the DMZ host *plato*: 172.016.114.010. This host was one of the hosts that were pursued in

the multi-staged attack of the LLD experiment. For this host, the fusion model analyzed 7 prioritized alerts for LLDOS 1.0 DMZ dataset of the sensor report. Without MLC, the alert fusion model clustered six *Sadmind_Amslverify_Overflow* alerts occurring within seconds of each other. However, conducting MLC did not cluster the remaining alert which was not similar to the other alerts clustered. Charts 1 and 2 in Figure 4.21 show the cluster coverage and cluster overall similarity for the host *plato* with and without MLC, for the RealSecure-NCSU sensor report.

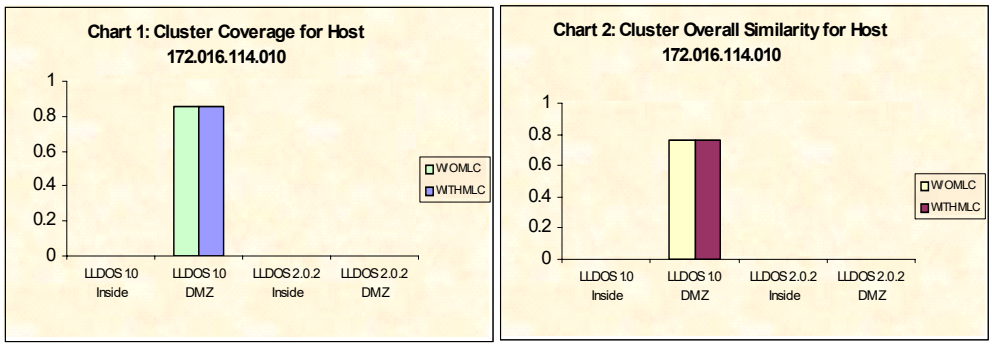


Figure 4.21 Cluster Coverage and Cluster Overall Similarity for Host *plato* analyzing the RealSecure-NCSU Sensor Report

Since, LLDOS 1.0 DMZ dataset was the only dataset to contain any prioritized alerts for this host, clusters could only be found for this dataset in the RealSecure-NCSU sensor report. Also, since the only type of exact cluster found was for alerts with the same features, both cluster coverage and cluster overall similarity remain the same with and without MLC (i.e., there is no additional improvement with MLC). This emphasizes that MLC is only helpful when there are variations in data.

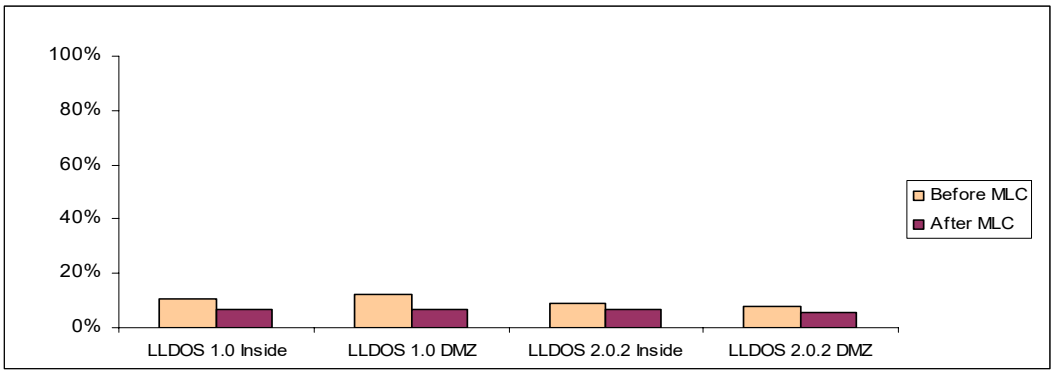


Figure 4.22 Alert Reduction with Multi-Level Clustering (MLC) for RealSecure-NCSU Sensor Report

Figure 4.22 shows the alert reduction performance of MLC for the RealSecure-NCSU sensor report. It shows that multi-level alert clustering further reduced alert volume after alert prioritization in terms of reporting only clustered alerts (i.e., alerts with common patterns).

*R&A for RealSecure-MSU Sensor Report*

In the case of LLDOS 1.0 Inside Zone dataset, with traditional exact clustering, the alert fusion model was not able to cluster any of the 31 prioritized alerts for the hosts. With MLC, the fusion model clustered 6 alerts for 3 hosts. In the case of LLDOS 1.0 DMZ dataset, without MLC, 3 alerts were clustered for 1 host out of 41 prioritized alerts. With MLC, the number of clustered alerts increased to 16 for 6 hosts. In the case of LLDOS 2.0.2 Inside Zone dataset, without MLC, none of 21 prioritized alerts were clustered. However, with MLC, 8 alerts were clustered for 3 hosts.

In the case of LLDOS 2.0.2 DMZ dataset, without MLC, the fusion model clustered only 2 alerts for 1 host out of the 15 alerts in the sensor report. With MLC, the number of clustered alerts increased to 4 for 2 hosts. Figure 4.23 shows the cluster coverage results with and without MLC for all four datasets in the RealSecure-MSU report.

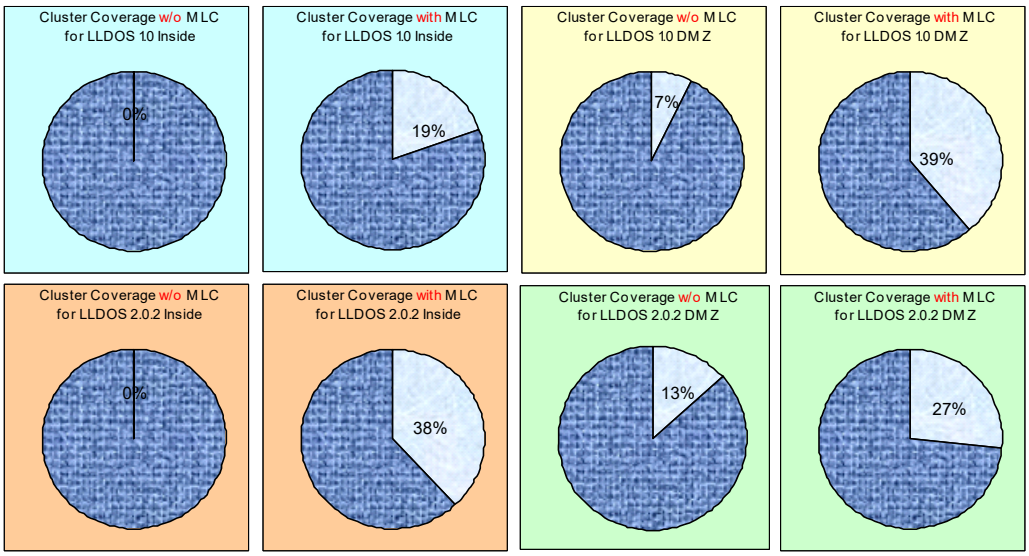


Figure 4.23 Comparison of Cluster Coverage with and without MLC for all Datasets in the RealSecure-MSU Sensor Report

The charts in Figure 4.23 show improvement in cluster coverage with MLC. The reason that cluster coverage, even with MLC, was not very high in this case is because the sensor reports for these datasets contained a large number of *Finger_User* alerts, which were isolated (i.e., they were generated for different targets coming from different sources at different times) but dominated approximately half of the sensor reports. As a result of this distribution, cluster coverage was not found to be high. Table 4.15 lists the

hosts for which alert clusters were found when analyzing the datasets in the RealSecure-MSU sensor report.

Table 4.15 Cluster Association Assessment for the RealSecure-MSU Sensor Report

| Dataset | Host | Cluster Association Strength (CAS) | Dataset | Host | Cluster Association Strength (CAS) |
|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | **172.016.112.010** | **65.51** | LLDOS 1.0 DMZ | **172.016.115.020** | **80.86** |
| | **172.016.112.050** | **65.51** | | **172.016.112.050** | **65.51** |
| | **172.016.115.020** | **65.51** | | **172.016.114.010** | **65.51** |
| | | | | **172.016.114.020** | **65.51** |
| | | | | **172.016.114.030** | **65.51** |
| | | | | 172.016.114.050 | 65.51 |
| LLDOS 2.0.2 Inside Zone | **172.016.112.050** | **65.51** | LLDOS 2.0.2 DMZ | **172.016.115.020** | **69.00** |
| | **172.016.115.020** | **65.51** | | 172.016.112.207 | 65.51 |
| | 172.016.112.207 | 65.51 | | | |

The following are interesting observations made from the results in Table 4.15:

In the case of LLDOS 1.0 Inside Zone dataset, only three hosts were reported with the same moderate CAS. These hosts were the actual victim hosts in the LLD experiment that were compromised using the same set of attacks. The reason the CAS is the same for all of them is that for each of these hosts, the alert fusion model found the following cluster:

- One *SameSource_SameAttack_RecentTime* type cluster consisting of two *Sadmind_Amslverify_Overflow* alerts occurring at similar time (i.e., within minutes).

Table 4.16 shows the CAS computed for these hosts for LLDOS 1.0 Inside Zone dataset along with the clusters generated and strength of the clusters.

Table 4.16 Cluster Association for LLDOS 1.0 Inside Zone Dataset analyzing the RealSecure-MSU Sensor Report

| Dataset | Host | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 172.016.112.010 | *SameSource_SameAttack_RecentTime* | 0.801 | 65.51 |
| | 172.016.112.050 | *SameSource_SameAttack_RecentTime* | 0.801 | |
| | 172.016.115.020 | *SameSource_SameAttack_RecentTime* | 0.801 | |

One of these victim hosts is *mill*: 172.016.115.020 and Table 4.15 shows that, in case of all the datasets, this host is reported with the highest CAS and for LLDOS 1.0 DMZ dataset, it is the highest. There were 8 prioritized alerts for this host in the case of LLDOS 1.0 DMZ dataset. Without MLC, the fusion model clustered three of them with the same alert features. With MLC, three additional alerts were clustered. The two clusters found were:

- One *SameSource_SameAttack_SameTime* type cluster with three *Sadmind_Amslverify_Overflow* alerts that occurred at the same time coming from the same source.
- One *SameSource_SameAttack_RecentTime* type cluster consisting of three additional *Sadmind_Amslverify_Overflow* alerts that occurred within minutes of the same alerts previously clustered.

Table 4.17 shows the CAS computed for the host *mill* analyzing the four datasets in the RealSecure-MSU sensor report, along with their clusters and the strength of the clusters.

Table 4.17 Cluster Association of Host *mill* for RealSecure-MSU Sensor Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource SameAttack RecentTime* | 0.801 | 65.51 |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 80.86 |
| | *SameSource SameAttack RecentTime* | 0.801 | |
| LLDOS 2.0.2 Inside Zone | *SameSource_SameAttack_RecentTime* | 0.801 | 65.51 |
| LLDOS 2.0.2 DMZ | *SameSource SameAttack SameTime* | 0.894 | 69.00 |

Here, the CAS is found to be highest for LLDOS 1.0 DMZ dataset because this dataset contained more variations of related alerts (hence more similar clusters) than did the other datasets. Table 4.17 also shows that for LLDOS 1.0 Inside Zone dataset and for LLDOS 2.0.2 Inside Zone and DMZ datasets only single clusters were found. However the cluster strengths were found to be different and the CAS reported for LLDOS 1.0 and LLDOS 2.0.2 Inside Zone datasets is the lowest among the datasets. This is because the cluster found for LLDOS 2.0.2 DMZ dataset involved no generalization and consisted only of alerts having the same features. Therefore, higher intra-cluster similarity or cluster strength contributed to a higher CAS than found for the ones having similar features (like in LLDOS 1.0 and LLDOS 2.0.2 Inside Zone datasets). Figure 4.24 compares the cluster coverage and cluster overall similarity for this host with and without MLC for all four datasets in the RealSecure-MSU sensor report.

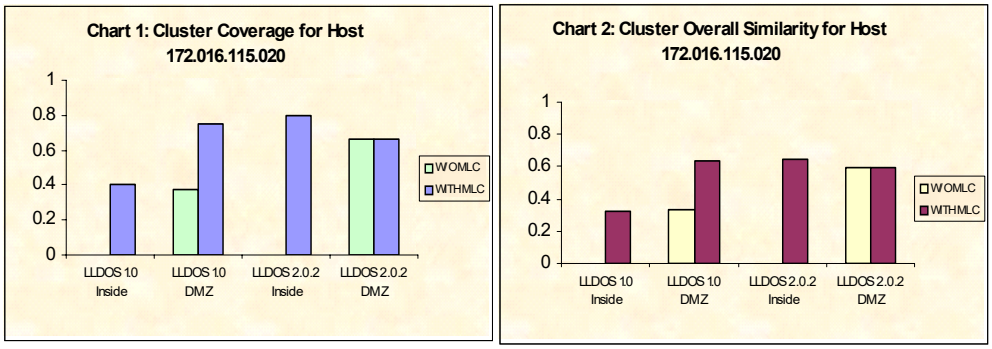


Figure 4.24 Cluster Coverage and Cluster Overall Similarity for Host *mill* analyzing the RealSecure-MSU Sensor Report

It should be noted that for LLDOS 2.0.2 DMZ dataset, where the only cluster found was for alerts with the same features, cluster coverage and cluster overall similarity both remain the same with and without MLC (i.e., there is no additional improvement with MLC). In contrast, for LLDOS 1.0 and LLDOS 2.0.2 Inside Zone datasets, since the alert fusion model did not cluster any alerts without MLC, the improvement with MLC is notable. For LLDOS 1.0 DMZ dataset, MLC clustered twice the number of alerts than without MLC.

Apart from the victim hosts, the only host that appears in Table 4.15 is the inside host *robin*: 172.016.112.207, for which a moderate CAS (65.50%) is reported for LLDOS 2.0.2 Inside and DMZ datasets. For both of these datasets, without MLC, the alert fusion model did not cluster any alerts. However, with MLC, one cluster was found, which is:

– *SameSource_SameAttack_RecentTime* type cluster with two *Finger_User* alerts generated at similar times (i.e., within minutes).

Charts 1 and 2 in Figure 4.25 show the cluster coverage and cluster overall similarity for this host with and without MLC for all four datasets in the RealSecure-MSU report.

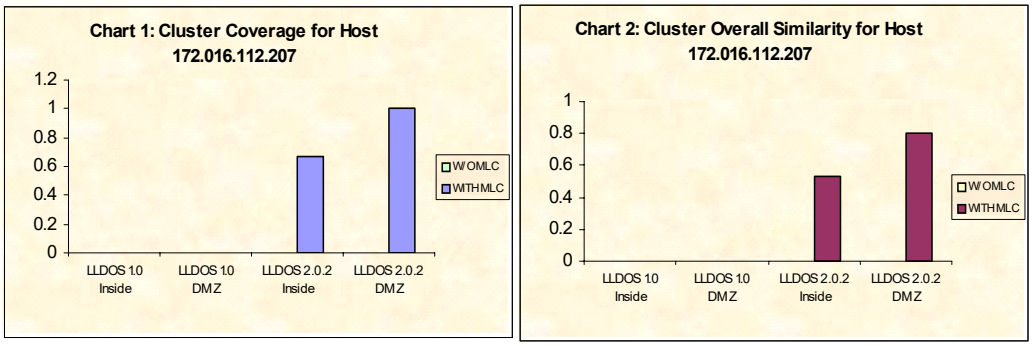


Figure 4.25 Cluster Coverage and Cluster Overall Similarity for Host *robin* analyzing the RealSecure-MSU Sensor Report

The charts show that for LLDOS 2.0.2 Inside and DMZ datasets, there was major improvement in cluster coverage and cluster overall similarity with MLC. Although the same clusters were found for both datasets, the results are better for the DMZ dataset as compared to the Inside Zone dataset because for LLDOS 2.0.2 DMZ dataset, all prioritized alerts were clustered. No alerts were clustered for LLDOS 1.0 Inside Zone and DMZ datasets (LLDOS 1.0 Inside Zone dataset contained only one prioritized alert and LLDOS 1.0 DMZ dataset did not contain any alerts for this host).

Figure 4.26 shows alert reduction performance of MLC for the RealSecure-MSU sensor report. It shows that multi-level clustering further reduced alert volume after alert prioritization in terms of reporting only clustered alerts (i.e., alerts with common patterns).

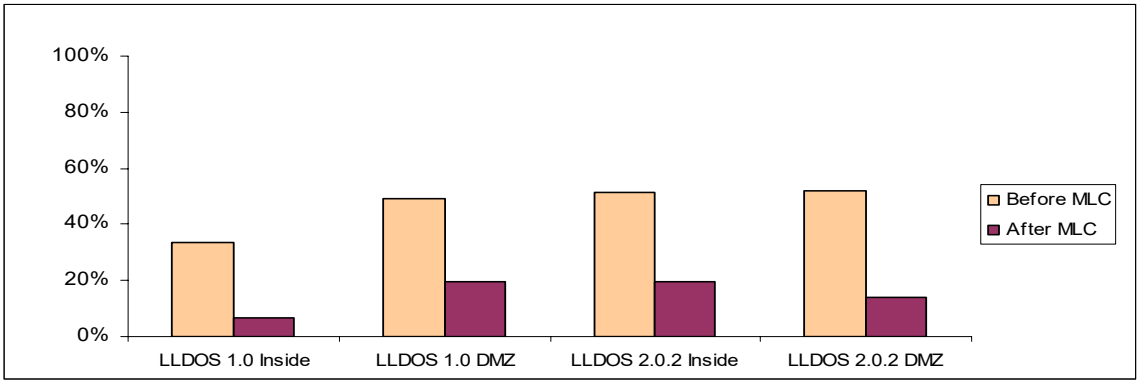


Figure 4.26 Alert Reduction with Multi-Level Clustering (MLC) for RealSecure-MSU Sensor Report

*R&A for Snort-MSU Sensor Report*

For LLDOS 1.0 Inside Zone dataset, with traditional exact clustering (i.e., without MLC), the fusion model clustered 52 alerts for 7 hosts out of 153 prioritized alerts. However, with MLC, the fusion model was able to cluster 72 additional alerts for 8 additional hosts (i.e., 124 alerts for 15 hosts). For LLDOS 1.0 DMZ dataset, without MLC, the fusion model clustered 107 alerts for 14 hosts out of 182 prioritized alerts. With MLC, number of clustered alerts increased to 149 for 17 hosts. For LLDOS 2.0.2 Inside Zone dataset, without MLC, out of 63 prioritized alerts, 19 alerts were clustered for 6 hosts. With MLC, 33 additional alerts were clustered, with a total of 96 alerts for 11 hosts. For LLDOS 2.0.2 DMZ dataset, without MLC, the alert fusion model clustered 21 alerts for 8 hosts out of 39 prioritized alerts. With MLC, the number of clustered alerts increased to 32 for 9 hosts. Figure 4.27 shows cluster coverage results with and without MLC for all four datasets in the Snort-MSU sensor report.

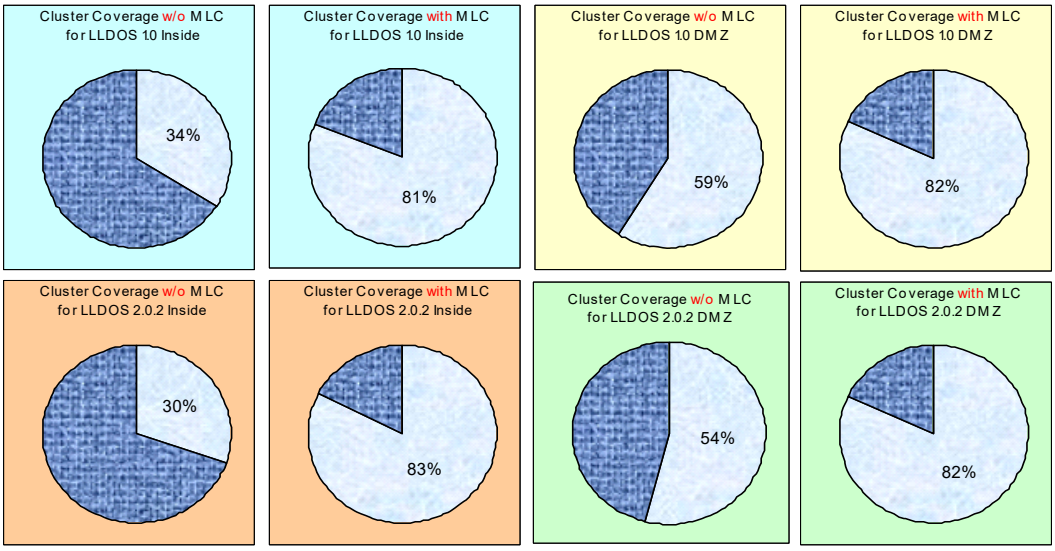


Figure 4.27 Comparison of Cluster Coverage with and without MLC for all Datasets in the Snort-MSU Sensor Report

The charts in Figure 4.27 show that with MLC, the alert fusion model was able to cluster more alerts that were related than without MLC. Table 4.18 lists the hosts, for which alert clusters were found analyzing the Snort-MSU sensor report, with their cluster association strengths.

Table 4.18 Cluster Association Assessment for Snort-MSU Sensor Report

| Dataset | Host | Cluster Association Strength (CAS) | Dataset | Host | Cluster Association Strength (CAS) |
|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 172.016.112.194 | 93.55 | LLDOS 1.0 DMZ | 172.016.114.050 | 83.95 |
| | **172.016.112.050** | **88.42** | | **172.016.112.050** | **80.86** |
| | 172.016.113.169 | 87.54 | | 172.016.112.149 | 80.86 |
| | 172.016.113.204 | 87.54 | | 172.016.112.194 | 80.86 |
| | 172.016.113.148 | 80.86 | | 172.016.113.084 | 80.86 |
| | 172.016.113.207 | 77.43 | | 172.016.113.148 | 80.86 |
| | 172.016.113.168 | 75.94 | | 172.016.113.169 | 80.86 |
| | **172.016.112.010** | **69.00** | | 172.016.113.207 | 80.86 |
| | **172.016.115.020** | **69.00** | | 172.016.113.204 | 80.86 |
| | 172.016.112.149 | 65.51 | | 172.016.113.168 | 80.86 |
| | 172.016.113.105 | 65.51 | | **172.016.112.010** | **69.00** |
| | 172.016.113.084 | 65.51 | | **172.016.114.010** | **69.00** |
| | 172.016.114.050 | 65.51 | | **172.016.114.020** | **69.00** |
| | 172.016.112.207 | 65.51 | | **172.016.115.020** | **69.00** |
| | 172.016.112.100 | 63.32 | | **172.016.114.030** | **69.00** |
| LLDOS 2.0.2 Inside Zone | | | | 172.016.112.207 | 65.51 |
| | | | | 172.016.113.105 | 65.51 |
| | 172.016.112.100 | 93.27 | LLDOS 2.0.2 DMZ | 172.016.112.100 | 88.92 |
| | 172.016.113.148 | 80.86 | | 172.016.112.194 | 80.86 |
| | **172.016.115.020** | **80.86** | | 172.016.113.204 | 80.86 |
| | 172.016.113.168 | 80.48 | | 172.016.112.149 | 69.00 |
| | 172.016.113.050 | 78.72 | | 172.016.113.050 | 69.00 |
| | 172.016.112.149 | 77.43 | | **172.016.115.020** | **69.00** |
| | 172.016.112.194 | 75.94 | | 172.016.113.168 | 69.00 |
| | 172.016.113.204 | 75.94 | | 172.016.113.084 | 69.00 |
| | **172.016.112.050** | **69.00** | | 172.016.113.105 | 65.51 |
| | 172.016.113.169 | 65.51 | | | |
| | 172.016.113.084 | 62.44 | | | |

Interesting observations made from the results shown in Table 4.18 are presented below:

In the case of LLDOS 1.0 Inside Zone dataset, the highest CAS was reported for an inside host, *falcon*: 172.016.112.194. Although this host was not one of the victim hosts involved in the multi-staged attack, there were multiple numbers of alerts for this host in the sensor report indicating active telnet communications to this host. This particular set of alerts might not have been malicious, however, the same type of communication can be indicative of intruders transferring files to a target host in preparation for an attack. For this host, the fusion model analyzed 18 prioritized alerts from the sensor report. Among them, without MLC, nine alerts were clustered into two *SameSource_ SameAttack_SameTime* type clusters. One such cluster consisted of three *ATTACK-RESPONSES_directory_listing* alerts and the other consisted of six *TELNET_access* alerts, which matched exactly in alert features. With MLC, the alert fusion model clustered 8 additional alerts that "somewhat" matched in alert features. The additional clusters found were:

- A *SimilarSource_SameAttack_SameTime* type cluster consisting of a *TELNET_access* alert that originated from the same network address range (172.016.xxx.xxx) and occurred within seconds of the same alerts from the same source.
- Two *SameSource_SameAttack_RecentTime* type clusters. One of them clustered three *TELNET_access* alerts originating from the source 172.016.112.050 at a similar time (i.e., within minutes). The other one consisted of three *TELNET_access* alerts originating from the source 135.008.060.182.

– A *SameSource_SimilarAttack_SameTime* type cluster consisting of *TELNET_access* alert that originated from 172.016.112.100 and occurred within seconds of a similar *ATTACK-RESPONSES_directory_listing* type of alerts from the same source. The alerts are similar because both are of the *Active_Communication* type.

Table 4.19 shows the CAS computed for host *falcon* analyzing all the datasets in the Snort-MSU sensor report along with the clusters and strength of the clusters.

Table 4.19 Cluster Association of Host *falcon* for Snort-MSU Sensor Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 93.54 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| | *SameSource_SimilarAttack_SameTime* | 0.801 | |
| | *SimilarSource_SameAttack_SameAttack* | 0.739 | |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 80.86 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| LLDOS 2.0.2 Inside Zone | *SameSource_SameAttack_RecentTime* | 0.801 | 75.94 |
| | *SimilarSource_SameAttack_RecentTime* | 0.688 | |
| LLDOS 2.0.2 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 80.86 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |

Table 4.19 shows the highest reported CAS for LLDOS 1.0 Inside Zone dataset because LLDOS 1.0 Inside Zone dataset contained more similar alert clusters than did the others. Chart 1 and 2 in Figure 4.28 compare the cluster coverage and cluster overall similarity for host *falcon* with and without MLC for all four datasets in the Snort-MSU sensor report.

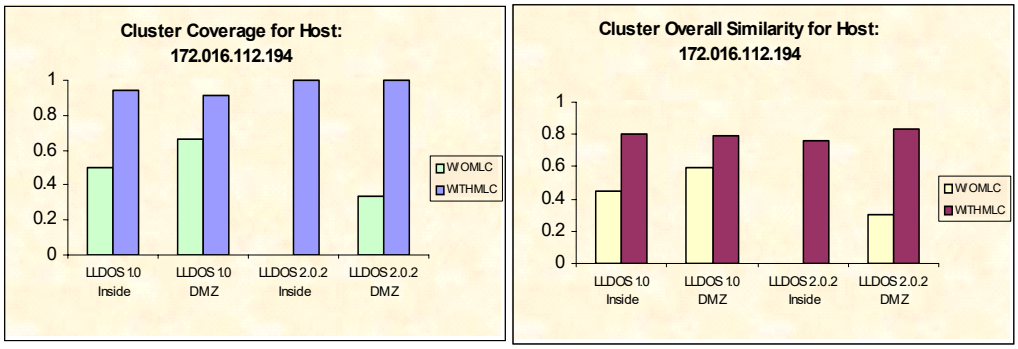


Figure 4.28 Cluster Coverage and Cluster Overall Similarity for Host *falcon* analyzing Snort-MSU Sensor Report

It should be noted that for LLDOS 2.0.2 Inside Zone dataset, no alerts were clustered without MLC. The charts show improvement in both cluster coverage and cluster overall similarity with MLC.

In the case of LLDOS 1.0 DMZ dataset, a moderate CAS (69.0%) was reported for a majority of the inside hosts that were actually pursued by the attacker in the multi-staged attack. Since the sensor report contained similar alerts for all of them, the same set of alert clusters were found for each. For example, for the inside victim host *locke*: 172.016.112.010, there were 10 prioritized alerts. From these, the fusion model clustered eight without MLC. The only type of cluster found was:

- Two *SameSource_SameAttack_SameTime* type. One consisting of four *RPC sadmind UDP NETMGT_PROC_SERVICE_overflow_attempt* alerts and one with four *RPC_sadmind_quer_with_root_credentials_attempt_UDP* alerts, all from the host: 202.077.162.213.

Other victim hosts with same CAS show similar results for this dataset. Table 4.20 shows the CAS computed for host *locke* analyzing the Snort-MSU sensor report along with the clusters generated and strength of the clusters.

Table 4.20 Cluster Association of Host *locke* for the Snort-MSU Sensor Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 69.00 |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 69.00 |
| LLDOS 2.0.2 Inside Zone | *None* | — | — |
| LLDOS 2.0.2 DMZ | *None* | — | — |

For this host, there were no alerts in the sensor report for LLDOS 2.0.2 Inside Zone and DMZ datasets. Chart 1 and 2 of Figure 4.29 compare the cluster coverage and cluster overall similarity for host *locke* with and without MLC for all four datasets in the Snort-MSU sensor report.

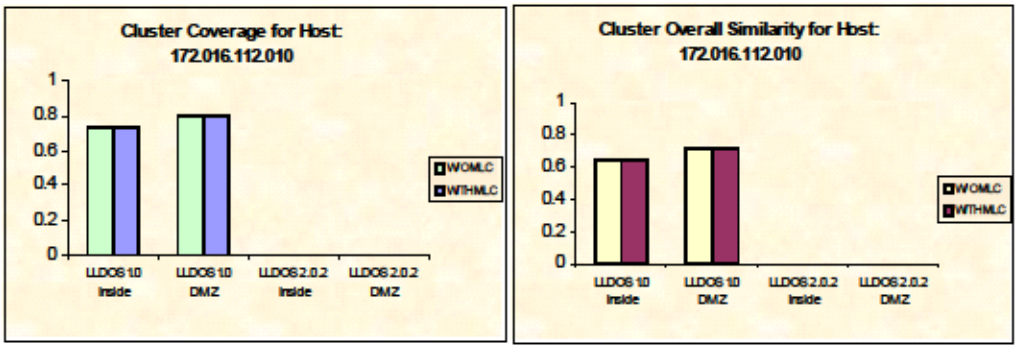


Figure 4.29 Cluster Coverage and Cluster Overall Similarity for Host *locke* analyzing the Snort-MSU Sensor Report

It should be noted that for LLDOS 1.0 Inside Zone and DMZ datasets, since the only type of exact cluster found was for alerts with the same features, cluster coverage and cluster overall similarity both remain the same with and without MLC.

In the case of LLDOS 2.0.2 Inside Zone dataset, among the actual victim hosts reported, Table 4.18 shows the highest CAS for host *mill*: 172.016.115.020. There were 8 prioritized alerts for this host in the sensor report. Without MLC, the fusion model clustered four of them. Using MLC, four more were clustered. The clusters found were:

- Two *SameSource_SameAttack_SameTime* type clusters. One cluster consisting of two *RPC_ sadmind_UDP_NETMGT_PROC_SERVICE_overflow_attempt* alerts and one cluster with two *RPC_sadmind_quer_with_ root_credentials_attempt_UDP* alerts.

- Two *SameSource_SameAttack_RecentTime* type clusters. One of them clustered two *TELNET_access* alerts generated from the source 172.016.112.050, occurring at similar time (i.e., within minutes). The other one consisted of two *DDOS_mstream_agent_to_handler* alerts, which were also generated from the same source.

Table 4.21 shows the CAS computed for host *mill* analyzing the Snort-MSU sensor report along with the clusters generated and strength of the clusters.

Table 4.21 Cluster Association of Host *mill* for the Snort-MSU Sensor Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 69.00 |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 69.00 |
| LLDOS 2.0.2 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 80.87 |
| | *SameSource_SameAttack_RecentTime* | 0.688 | |
| LLDOS 2.0.2 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 69.00 |

Here the CAS is found to be highest for LLDOS 2.0.2 Inside Zone dataset because this dataset contained more similar clusters than did the others. Charts 1 and 2 of Figure 4.30 compare the cluster coverage and cluster overall similarity for host *mill* with and without MLC for all four datasets in the Snort-MSU sensor report.

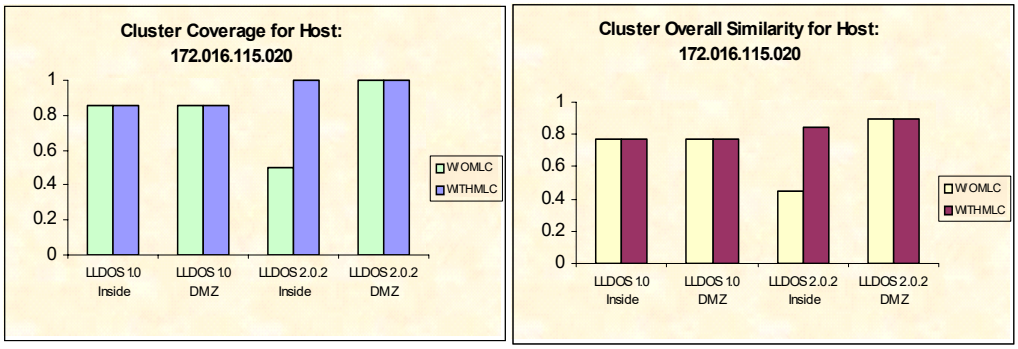


Figure 4.30 Cluster Coverage and Cluster Overall Similarity for Host *mill* analyzing the Snort-MSU Sensor Report

It should be noted that for all datasets except the LLDOS 2.0.2 Inside Zone dataset, cluster coverage and cluster overall similarity both remain the same with and without MLC. This is because the only type of exact cluster found was for alerts with the same features. The improvement in cluster coverage and cluster overall similarity is apparent for LLDOS 2.0.2 Inside Zone dataset, where there were variations in data and hence the MLC technique yielded more clusters consisting of alerts with similar features.

In the case of LLDOS 2.0.2 DMZ dataset, the lowest CAS is reported for the host *swallow*: 172.016.113.105. There were 2 prioritized alerts analyzed for this host. Without MLC, the fusion model did not cluster any. However, with MLC, the fusion model was able to cluster two alerts with similar features. The only cluster found was:

– *SameSource_SameAttack_RecentTime* type with two *Telnet_Access* alerts that were generated within minutes of each other.

Table 4.22 shows the CAS computed for host *swallow* analyzing the Snort-MSU sensor report along with the clusters generated and strength of the clusters.

Table 4.22 Cluster Association of Host *swallow* for the Snort-MSU Sensor Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_RecentTime* | 0.801 | 65.50 |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_RecentTime* | 0.801 | 65.50 |
| LLDOS 2.0.2 Inside Zone | *None* | — | — |
| LLDOS 2.0.2 DMZ | *SameSource_SameAttack_RecentTime* | 0.801 | 65.50 |

In the case of LLDOS 2.0.2 Inside Zone dataset, no alerts were clustered since there were only 2 prioritized alerts that were not similar. For all other datasets, the cluster strengths were the same because the only clusters found were generated at the same level of abstraction (level 3). Charts 1 and 2 in Figure 4.31 show the cluster coverage and cluster overall similarity for this host with and without MLC for all four datasets in the Snort-MSU sensor report.

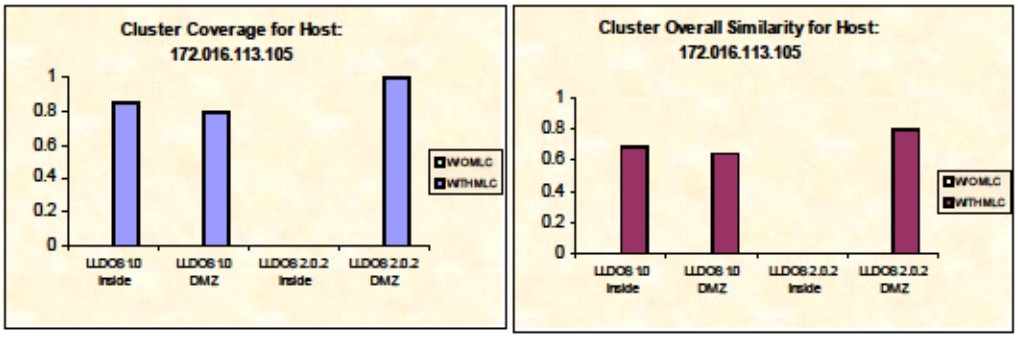


Figure 4.31 Cluster Coverage and Cluster Overall Similarity for Host *swallow* analyzing the Snort-MSU Sensor Report

It can be seen from the charts that with MLC there was major improvement in cluster coverage and cluster overall similarity. The results are best for LLDOS 2.0.2 DMZ dataset where all prioritized alerts were clustered.

Figure 4.32 shows the alert reduction performance of MLC for the Snort-MSU sensor report. It shows that multi level clustering further reduced alert volume after alert prioritization in terms of reporting only clustered alerts (i.e., alerts with common patterns).

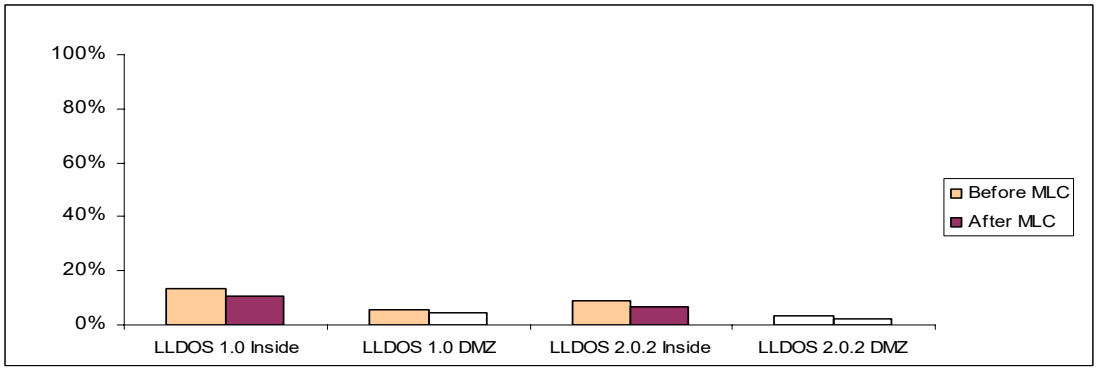


Figure 4.32 Alert Reduction with Multi-Level Clustering (MLC) for Snort-MSU Sensor Report

*R&A for MultiSensor-MSU Report*

In the case of LLDOS 1.0 Inside Zone dataset, with traditional exact clustering, the fusion model clustered 52 alerts for 7 hosts from 184 prioritized alerts. However with MLC, the fusion model was able to cluster 88 additional alerts for 8 additional hosts (for a total of 180 alerts for 15 hosts). In the case of LLDOS 1.0 DMZ dataset, without MLC, the fusion model clustered 110 alerts for 14 hosts out of 223 prioritized alerts. With MLC, the number of clustered alerts increased to 177 for 17 hosts. In the case of LLDOS

2.0.2 Inside Zone dataset, without MLC, 19 alerts were clustered for 6 hosts out of 84 prioritized alerts. With MLC, 43 additional alerts were clustered for 6 additional hosts. In the case of LLDOS 2.0.2 DMZ dataset, without MLC, the alert fusion model clustered 23 alerts for 8 hosts out of 54 prioritized alerts. With MLC, number of clustered alerts increased to 36 for 10 hosts. Figure 4.33 shows the cluster coverage results with and without MLC for all four datasets in the MultiSensor-MSU report.

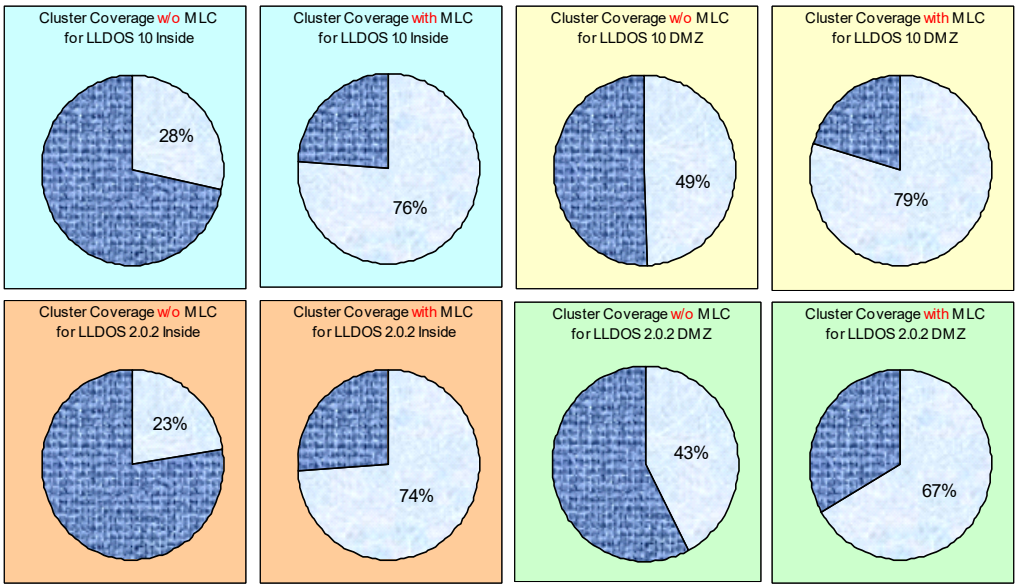


Figure 4.33 Comparison of Cluster Coverage with and without MLC for all Datasets in the MultiSensor-MSU Report

The charts in the figure show that cluster coverage is notably better for MLC than without MLC. Table 4.23 lists the hosts for which alert clusters were found analyzing the MultiSensor-MSU report, with their cluster association strengths.

Table 4.23 Cluster Association Assessment for MultiSensor-MSU Report

| Dataset | Host | Cluster Association Strength (CAS) | Dataset | Host | Cluster Association Strength (CAS) |
|---|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | **172.016.112.050** | **93.55** | LLDOS 1.0 DMZ | 172.016.114.050 | 91.52 |
| | 172.016.112.194 | 93.55 | | **172.016.112.050** | **88.92** |
| | **172.016.112.010** | **88.92** | | **172.016.114.010** | **88.92** |
| | **172.016.115.020** | **88.92** | | **172.016.114.030** | **88.92** |
| | 172.016.113.169 | 88.42 | | **172.016.114.020** | **88.92** |
| | 172.016.113.204 | 87.54 | | **172.016.115.020** | **88.92** |
| | 172.016.113.168 | 85.08 | | **172.016.112.010** | **80.86** |
| | 172.016.113.148 | 80.86 | | 172.016.112.149 | 80.86 |
| | 172.016.113.207 | 77.43 | | 172.016.113.084 | 80.86 |
| | 172.016.112.100 | 76.62 | | 172.016.113.148 | 80.86 |
| | 172.016.112.149 | 65.51 | | 172.016.113.169 | 80.86 |
| | 172.016.112.207 | 65.51 | | 172.016.113.207 | 80.86 |
| | 172.016.114.050 | 65.51 | | 172.016.113.204 | 80.86 |
| | 172.016.113.084 | 65.51 | | 172.016.113.168 | 80.86 |
| | 172.016.113.105 | 65.51 | | 172.016.112.194 | 80.86 |
| LLDOS 2.0.2 Inside Zone | | | | 172.016.112.207 | 65.51 |
| | | | | 172.016.113.105 | 65.51 |
| | 172.016.112.100 | 93.27 | LLDOS 2.0.2 DMZ | 172.016.112.100 | 88.92 |
| | **172.016.112.050** | **88.92** | | 172.016.112.194 | 80.86 |
| | 172.016.112.194 | 85.08 | | 172.016.113.204 | 80.86 |
| | 172.016.113.148 | 80.86 | | 172.016.112.149 | 69.00 |
| | **172.016.115.020** | **80.86** | | 172.016.113.084 | 69.00 |
| | 172.016.113.168 | 80.48 | | **172.016.115.020** | **69.00** |
| | 172.016.113.050 | 80.08 | | 172.016.113.168 | 69.00 |
| | 172.016.112.149 | 77.43 | | 172.016.113.050 | 69.00 |
| | 172.016.113.204 | 77.43 | | 172.016.112.207 | 65.51 |
| | 172.016.113.084 | 75.02 | | 172.016.113.105 | 65.51 |
| | 172.016.112.207 | 65.51 | | | |
| | 172.016.113.169 | 65.51 | | | |

Interesting observations made from the results in Table 4.23 are presented below:

In the case of LLDOS 1.0 Inside Zone dataset, one of the highest CAS was reported for the compromised host *pascal*: 172.016.112.050, for which the alert fusion model analyzed 18 prioritized alerts from RealSecure and Snort. Among them, without MLC,

the fusion model clustered six alerts in two *SameSource_SameAttack_SameTime* type clusters. One of such cluster consisted of three *RPC_sadmind_query_with_root_credentials_ attempt_UDP* alerts and the other one consisted of three *RPC_sadmind_UDP_ NETMGT_PROC_SERVICE_overflow_attempt* alerts. All of these alerts were Snort generated and matched exactly in their alert features. With MLC, the fusion model clustered ten additional alerts that matched "somewhat" in alert features. The different types of clusters found were:

- Two *SameSource_SimilarAttack_SameTime* type clusters consisting of two alerts each. One cluster consisted of the alerts: *Sadmind_Ping* and *RPC_Sadmind_ UDP_Ping*, reported by the sensors RealSecure and Snort, respectively. The fusion model found these alerts similar because both are *Probe_of_Service* type. Another same type cluster consisted of the alerts: *DDOS_mstream_handler_ to_agent* and *Mstream_zombie_request*, reported by RealSecure and Snort, respectively. The fusion model clustered these similar alerts because both belong to the *Launch_Importation* category.

- Three *SameSource_SameAttack_RecentTime* type clusters. One of these clustered two *Sadmind_Amslverify_Overflow* alerts generated from RealSecure, which occurred at similar times (i.e., within minutes). Because of the close time proximity, each of the other two similar clusters associated one additional Snort alert (which are *RPC_sadmind_query_with_root_credentials_ attempt_UDP* and *RPC_sadmind_UDP_NETMGT_PROC_SERVICE_overflow_ attempt* alerts) with previously found exact clusters of the same alert.

- One *SimilarSource_SameAttack_SameTime* type cluster that consisted of two Snort *Telnet_access* alerts originating from the same network address range (i.e., 172.016.xxx.xxx).

Table 4.24 shows the CAS computed for host *pascal* analyzing the MultiSensor-MSU report along with the clusters generated and their strengths.

Table 4.24 Cluster Association of Host *pascal* for the MultiSensor-MSU Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 93.54 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| | *SameSource_SimilarAttack_SameTime* | 0.801 | |
| | *SimilarSource_SameAttack_SameTime* | 0.739 | |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 88.92 |
| | *SameSource_SameAttack_RecentTime* | 0.834 | |
| | *SameSource_SimilarAttack_SameTime* | 0.834 | |
| LLDOS 2.0.2 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 88.92 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| | *SameSource_SimilarAttack_SameTime* | 0.801 | |
| LLDOS 2.0.2 DMZ | *None* | — | — |

Table 4.24 shows the CAS computed for host *pascal* to be higher for LLDOS 1.0 Inside Zone dataset than for LLDOS 1.0 DMZ dataset and LLDOS 2.0.2 Inside Zone dataset because LLDOS 1.0 Inside Zone dataset contained more similar clusters. Charts 1 and 2 in Figure 4.34 compare the cluster coverage and cluster overall similarity for this victim host with and without MLC for the MultiSensor-MSU report.

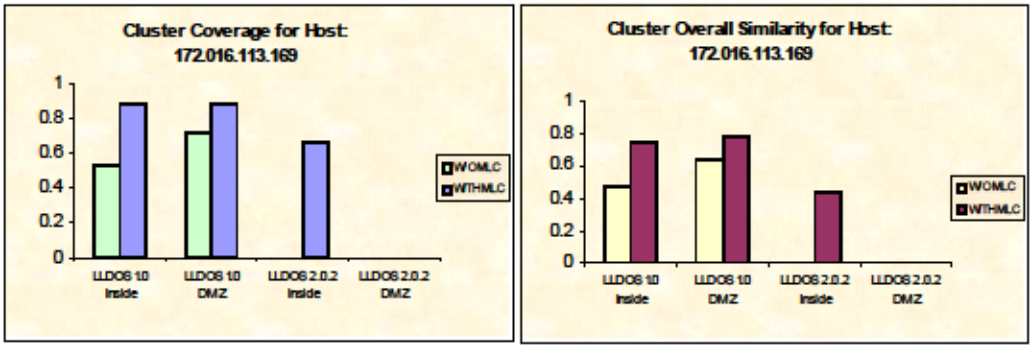


Figure 4.34 Cluster Coverage and Cluster Overall Similarity for Host *pascal* analyzing the MultiSensor-MSU Report

It should be noted that for LLDOS 2.0.2 DMZ dataset, no alerts were clustered for this host because the only two applicable alerts in LLDOS 2.0.2 DMZ dataset were not similar to each other. The charts show improved cluster coverage and cluster overall similarity using MLC.

In the case of LLDOS 1.0 DMZ dataset, the highest CAS was reported for one of the DMZ hosts, *marx*: 172.016.114.050, for which the fusion model analyzed 13 prioritized alerts from RealSecure and Snort. After analyzing this dataset without MLC, none of the alerts were clustered for this host. However, with MLC, the fusion model was able to cluster nine alerts with similar alert features. The clusters found were:

- One *SameSource_SameAttack_RecentTime* type cluster with two *ICMP_flood* alerts from RealSecure that were generated in similar times (i.e., within minutes).

- One *SameSource_SimilarAttack_SameTime* type cluster with two alerts, *WEB-MISC_/doc/_access* and *WEB-CGI_finger_access*, generated from the same source and at the same time. The fusion model clustered them because they were both found to be of the *Privilege_Violation* type.

- Two *SameSource_SimilarAttack_RecentTime* type clusters with two alerts each. One cluster grouped *Privilege_Violation* type of alerts: *WEBMISC_/doc/_access* and *WEB-MISC_backup_access*, which were reported originating from the host 197.218.177.069 within minutes of each other. The other cluster also consisted of these same alerts that come from host 194.007.248.153 within minutes of each other.

- One *SimilarSource_SameAttack_RecentTime* cluster that consisted of a *WEB-MISC_/doc/_access* alert, which originated from the same class C address range (194.xxx.xxx.xxx) from which the same alerts originated within minutes of each other.

Table 4.25 shows the CAS computed for the host *marx* analyzing the MultiSensor-MSU report along with the clusters generated and their strengths.

Table 4.25 Cluster Association of Host *marx* for the MultiSensor-MSU Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SimilarAttack_SameTime* | 0.801 | 65.50 |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_RecentTime* | 0.801 | 91.52 |
| | *SameSource_SimilarAttack_SameTime* | 0.801 | |
| | *SameSource_SimilarAttack_RecentTime* | 0.739 | |
| | *SimilarSource_SameAttack_RecentTime* | 0.632 | |
| LLDOS 2.0.2 Inside Zone | *None* | — | — |
| LLDOS 2.0.2 DMZ | *None* | — | — |

The CAS was found to be higher for LLDOS 1.0 DMZ dataset than for LLDOS 1.0 Inside Zone dataset because LLDOS 1.0 DMZ dataset contained more variations of related alerts (hence more similar clusters) than did LLDOS 1.0 Inside Zone dataset. No alerts were clustered for this host for LLDOS 2.0.2 Inside Zone and DMZ datasets because the alerts were not similar. Charts 1 and 2 in Figure 4.35 show the cluster coverage and cluster overall similarity for host *marx* with and without MLC for the MultiSensor-MSU report.

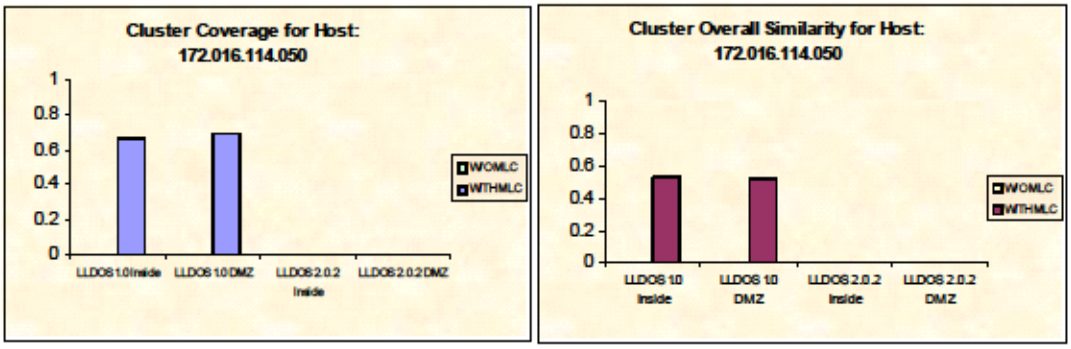


Figure 4.35 Cluster Coverage and Cluster Overall Similarity for Host *marx* analyzing the MultiSensor-MSU Report

It should be noted that since no clusters of alerts were found without MLC, cluster coverage and cluster overall similarity was zero for all datasets. Therefore, with MLC, there was improvement in cluster coverage and cluster overall similarity for LLDOS 1.0 Inside Zone and DMZ datasets. These cases again emphasize the usefulness of MLC when there are variations in data.

In the case of LLDOS 2.0.2 Inside Zone dataset, the lowest CAS was reported for the host *swan*: 172.016.113.169. There were 3 prioritized alerts analyzed for this host, generated by RealSecure and Snort. Without MLC, the alert fusion model did not cluster any alerts. However, with MLC, the fusion model was able to cluster two of the alerts with similar features. The only cluster found was:

- A *SameSource_SameAttack_RecentTime* type cluster with two *Telnet Access* alerts from Snort that were generated at similar times (i.e., within minutes).

Table 4.26 shows CAS computed for this host analyzing the MultiSensor-MSU report along with the clusters generated and the strength of the clusters.

Table 4.26 Cluster Association of Host *swan* for the MultiSensor-MSU Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 88.42 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| | *SimilarSource_SameAttack_SameTime* | 0.739 | |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 80.86 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| LLDOS 2.0.2 Inside Zone | *SameSource_SameAttack_RecentTime* | 0.801 | 65.50 |
| LLDOS 2.0.2 DMZ | *None* | — | — |

The CAS was found to be highest for LLDOS 1.0 Inside Zone dataset because it contained more similar clusters than did LLDOS 1.0 DMZ dataset or LLDOS 2.0.2 Inside Zone dataset. Charts 1 and 2 in Figure 4.36 show the cluster coverage and cluster overall similarity for this host with and without MLC for the MultiSensor-MSU report.

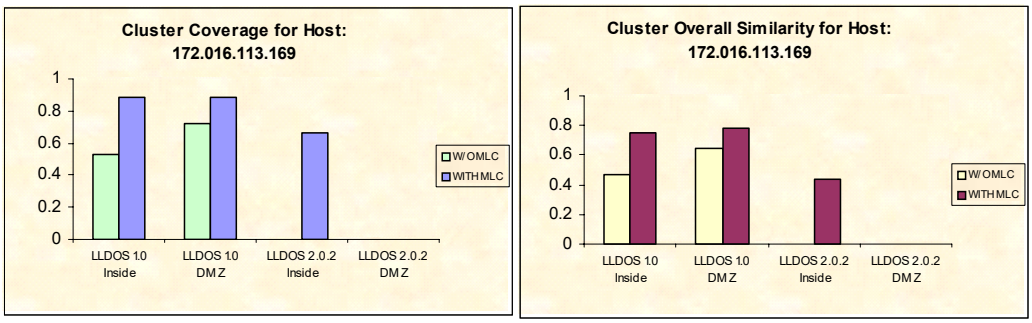


Figure 4.36 Cluster Coverage and Cluster Overall Similarity for Host *swan* analyzing the MultiSensor-MSU Report

The charts in Figure 4.36 show that in case of LLDOS 2.0.2 Inside Zone dataset, no alerts were clustered without MLC, therefore cluster coverage and cluster overall similarity is zero. With MLC, there was notable improvement in cluster coverage and cluster overall similarity. No alerts were clustered for this host for LLDOS 2.0.2 DMZ dataset, with or without MLC.

In the case of LLDOS 2.0.2 DMZ dataset, a moderate CAS was determined for one of the victim hosts, *mill*: 172.016.115.020. There were 7 prioritized alerts for this host from RealSecure and Snort. Without MLC, the fusion model clustered six of them because pairs of the alerts had the same features. Using MLC did not cluster any additional alerts because the only remaining alert had entirely different alert features than the others. The only cluster found was:

– Three *SameSource_SameAttack_SameTime* type clusters. One with two *Sadmind_Amslverify_Overflow* alerts from RealSecure, one with two *RPC sadmind UDP NETMGT_PROC_SERVICE_overflow_attempt* alerts from Snort and one with two *RPC_sadmind_quer_with_root_credentials_attempt_UDP* alerts from Snort.

Table 4.27 shows the CAS computed for host *mill* analyzing the MultiSensor-MSU report along with the clusters generated and their strengths.

Table 4.27 Cluster Association of Host *mill* for the MultiSensor-MSU Report

| Dataset | Cluster_Type | Cluster Strength (Intra-Cluster Similarity) | CAS |
|---|---|---|---|
| LLDOS 1.0 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 88.92 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| | *SameSource_SimilarAttack_SameTime* | 0.801 | |
| LLDOS 1.0 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 88.92 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| | *SameSource_SimilarAttack_SameTime* | 0.801 | |
| LLDOS 2.0.2 Inside Zone | *SameSource_SameAttack_SameTime* | 0.894 | 80.86 |
| | *SameSource_SameAttack_RecentTime* | 0.801 | |
| LLDOS 2.0.2 DMZ | *SameSource_SameAttack_SameTime* | 0.894 | 69.00 |

The CAS was found to be highest for LLDOS 1.0 Inside Zone and DMZ datasets because these datasets contained more variations of related alerts. Charts 1 and 2 of Figure 4.37 compare the cluster coverage and cluster overall similarity for this host with and without MLC for all of the four datasets in the MultiSensor-MSU report.

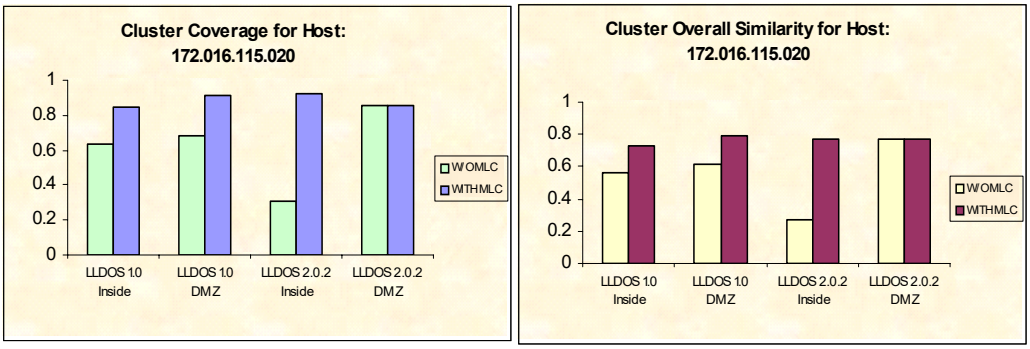


Figure 4.37 Cluster Coverage and Cluster Overall Similarity for Host *mill* analyzing the MultiSensor-MSU Report

It should be noted that in case of LLDOS 2.0.2 DMZ dataset, since the only type of exact cluster found was for alerts with the same features, cluster coverage and cluster overall similarity both remain the same with and without MLC (i.e., there was no additional improvement with MLC).

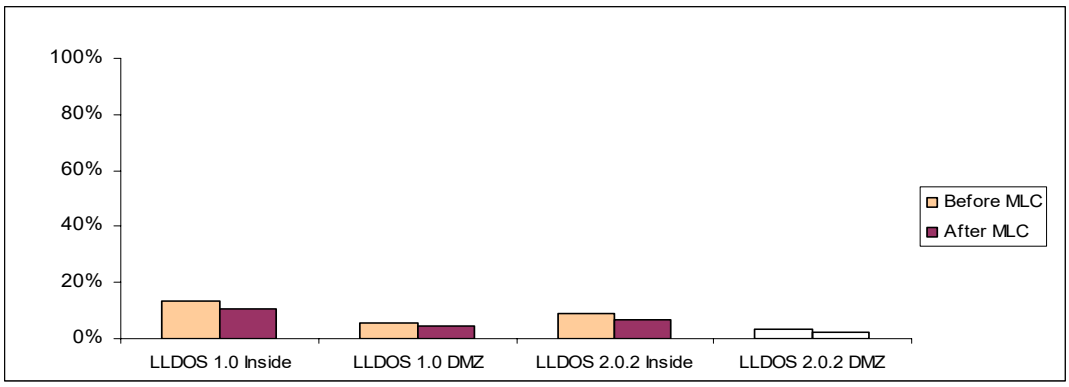


Figure 4.38 Alert Reduction with Multi-Level Clustering (MLC) for MultiSensor MSU Report

Figure 4.38 shows alert reduction before and after multi-level clustering of the MultiSensor-MSU report. It shows that multi-level clustering further reduced alert volume after alert prioritization in terms of reporting only clustered alerts (i.e., alerts with common patterns).

*Summary for Alert Clustering Experiment*

In the experiment conducted for multi-level alert clustering, we found that the fusion model was able to cluster alerts with common attack patterns. Results show that multi-level clustering was able to cluster more alerts with the same and similar features than did traditional clustering. In most cases, when there were variations in data, the improvement with multi-level clustering was apparent. In the worst case (i.e., when data did not contain variations in alert features), multi-level clustering performed the same as traditional alert clustering. The alert clusters were reported with a quantitative assessment of the strength of the clusters. The strength of the clusters depended on the similarity of alerts in the cluster. That is, a high value for cluster strength indicated the presence of alerts in the cluster whose features were more similar, if not the same, and a low value of cluster strength indicated the presence of alerts in the cluster whose features were less similar. For each host reported under attack, an overall degree of concern for cluster association was also reported. For example, a high cluster association strength reported for a host indicated that multiple alert clusters were found for that host, suggesting that the attacker(s) attempted different variations of the attacks on the host. A low cluster association strength reported for a host indicated that only single or less similar alert clusters was/were found for the host, implying that the attacks targeted at the hosts were

mostly dispersed. Thus cluster association strengths and incident strengths provided the security administrator with an insight into the extent of concern for hosts involved in common attack patterns.

### *4.3.4 Misuse Situation Assessment Experiment*

*Objective*

Situation assessment is the final task of the alert fusion model's higher-level reasoning process where the fusion model provides an overall security assessment for hosts. The purpose of the misuse situation assessment experiment was to evaluate the dynamic fusion technique that the fusion model employs to combine evidence of hosts' degree of involvement in multi-staged attacks and in common attack patterns. The experiment was conducted on four sensor alert reports - RealSecure-NCSU, RealSecure-MSU, Snort-MSU and MultiSensor-MSU individually.

*Results and Analysis (R&A)*

The following are the results and analysis of the situation assessment experiment for misuse sensor reports. It should be noted that misuse situation assessment occurs after abstract alert correlation and multi- level alert clustering and combines the results of the two.

*R&A for RealSecure-NCSU Sensor Report*

This experiment was conducted on the RealSecure-NCSU sensor report. Table 4.28 shows the situation assessment for reported hosts with their overall degree of concern (ODOC) - determined by fusing their incident association strengths (IAS)s and cluster association strengths (CAS)s. As discussed in section 3.2.3 of Chapter III, once a quantitative assessment of a host's or resource's involvement in anomalous or malicious situation is determined, a resource concern model can be used to report the results to the security administrator. In accordance with the resource concern model shown in Figure 3.13 of Chapter III, the reported hosts in Table 4.28 are color coded depending on the value of the ODOC (SEVERE: >80%; HIGH: >70% and <=80%; ELEVATED: >50% and <=70%; CAUTIOUS: >40% and <=50%; LOW: <=40%) to visually relate the results to the reader.

Table 4.28 Situation Assessment for the RealSecure-NCSU Sensor Report

| Dataset | Host | IAS | CAS | ODOC |
|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 172.016.112.050 | 91.78 | 72.88 | 84.19 |
| | 172.016.115.020 | 91.78 | 75.94 | 84.11 |
| | 172.016.112.010 | 91.78 | 65.51 | 73.15 |
| | 172.016.113.148 | 27.00 | 72.06 | 48.88 |
| | 172.016.112.100 | | 73.64 | 39.87 |
| | 172.016.112.149 | | 58.60 | 33.63 |
| | 172.016.112.194 | | 58.60 | 33.63 |
| | 172.016.113.084 | | 58.60 | 33.63 |
| | 172.016.113.168 | | 58.60 | 33.63 |
| | 172.016.113.169 | | 58.60 | 33.63 |
| | 172.016.113.204 | | 58.60 | 33.63 |
| LLDOS 1.0 DMZ | 172.016.115.020 | 52.06 | 75.94 | 62.13 |
| | 172.016.112.010 | 52.06 | 65.51 | 56.79 |
| | 172.016.112.050 | 52.06 | 65.51 | 56.79 |
| | 172.016.113.148 | 27.00 | 72.46 | 49.36 |
| | 172.016.114.020 | 27.00 | 65.51 | 47.31 |
| | 172.016.114.030 | 27.00 | 65.51 | 47.31 |
| | 172.016.114.010 | 27.00 | 69.00 | 45.23 |
| | 172.016.112.100 | | 74.62 | 40.85 |
| | 172.016.112.149 | | 58.60 | 33.63 |
| | 172.016.113.084 | | 58.60 | 33.63 |
| | 172.016.113.168 | | 58.60 | 33.63 |
| | 172.016.113.169 | | 58.60 | 33.63 |
| LLDOS 2.0.2 Inside Zone | 172.016.112.050 | 72.80 | 76.62 | 82.58 |
| | 172.016.115.020 | 72.80 | 75.94 | 82.48 |
| | 172.016.112.100 | | 75.94 | 42.07 |
| | 172.016.112.194 | | 65.51 | 40.41 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.084 | | 65.51 | 40.41 |
| | 172.016.113.105 | | 65.51 | 40.41 |
| | 172.016.113.148 | | 65.51 | 40.41 |
| | 172.016.113.169 | | 65.51 | 40.41 |
| | 172.016.113.204 | | 65.51 | 40.41 |
| LLDOS 2.0.2 DMZ | 172.016.115.020 | 34.37 | 75.94 | 53.72 |
| | 172.016.112.100 | | 65.51 | 40.41 |
| | 172.016.112.194 | | 65.51 | 40.41 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.084 | | 65.51 | 40.41 |
| | 172.016.113.105 | | 65.51 | 40.41 |
| | 172.016.113.148 | | 65.51 | 40.41 |
| | 172.016.113.169 | | 65.51 | 40.41 |
| | 172.016.113.204 | | 65.51 | 40.41 |

Misuse situation assessment with the RealSecure-NCSU sensor report yielded good results. Table 4.28 shows that the only hosts reported with SEVERE, HIGH or ELEVATED levels of concerns were the actual victim hosts of the LLD 2000 attacks experiments. CAUTIOUS or LOW level of concerns represent hosts, which were not compromised but for which there were evidence of some form of anomalous activities present in the sensor report that resulted in incidents activated or clusters generated or both. The following discusses examples from Table 4.28 that lend insight into how the dynamic fusion technique works in misuse situation assessment.

In the case of LLDOS 1.0 Inside Zone dataset, for one victim host, *pascal*: 172.016.112.050, the incident association was reported as 91.78% (IAS) and cluster association was reported to be 72.88% (CAS). Combining these resulted in an overall degree of concern of 84.19%. According to the resource concern model, this output signifies a SEVERE level of concern for the host. Figure 4.39 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

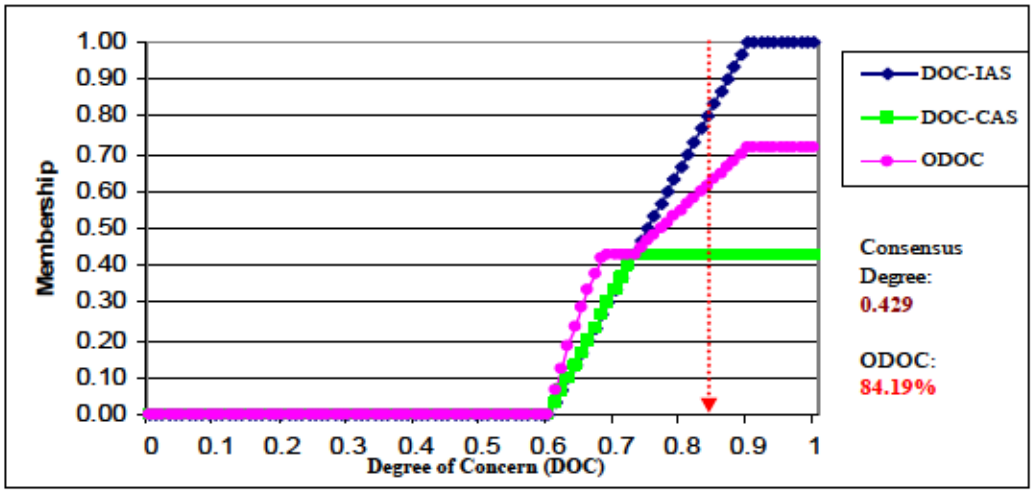


Figure 4.39 Dynamic Fusion Results for Host *pascal* for LLDOS 1.0 Inside Zone Dataset analyzing the RealSecure-NCSU Sensor Report

Both of the possibility distributions of the degree of concern related by IAS (DOC-IAS) and the degree of concern related by CAS (DOC-CAS) consist of *High* fuzzy sets and show weak conflict between them. As expected (explained in section 3.2.3 of Chapter III), the combined output fuzzy set shows additive behavior until the consensus degree or level of consensus (0.492) is reached and more conflict starts to arise between the inputs. From that point onwards, the output fuzzy set shows compromised behavior. The arrow points to the defuzzified output value in the *High* region.

In the case of LLDOS 2.0.2 DMZ dataset, for the inside host *goose*: 172.016.113.204, while cluster association was reported to be 65.51% (CAS), there was no report of incident association by the fusion model. This resulted in an overall degree of concern of 40.41%. According to the resource concern model, this output would signify a CAUTIOUS level of concern for the host. Figure 4.40 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

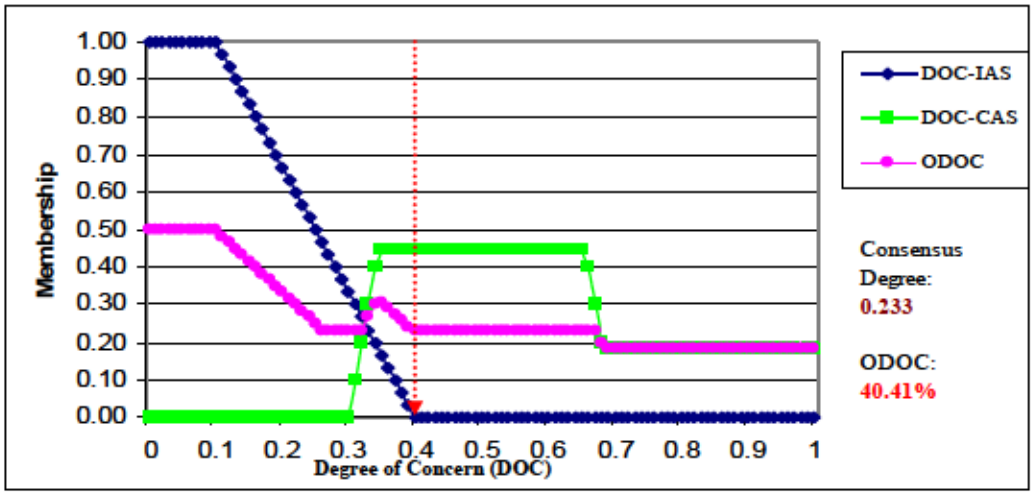


Figure 4.40 Dynamic Fusion Results for Host *goose* for LLDOS 2.0.2 DMZ Dataset analyzing the RealSecure-NCSU Sensor Report

The possibility distribution of DOC-IAS has membership in the *Low* fuzzy set and that of DOC-CAS has different memberships in the *Medium* and *High* fuzzy sets. Figure 4.40 shows strong conflict between them in the *Medium* fuzzy region and total conflict in the *Low* and *High* fuzzy regions. As expected, the combined output fuzzy set shows additive behavior within the level of consensus (0.233) and compromised behavior beyond it. The arrow points to the defuzzified output value at the lower end of the *Medium* region.

The results of situation assessment for this experiment shows that whenever there was strong agreement between the results reported by abstract alert correlation and multi-level alert clustering, the combined overall concern increased (e.g., in the case of LLDOS 2.0.2 Inside Zone dataset and for the host *mill*: 172.016.115.020, similar values for IAS and CAS were reported (72.80% and 75.94% respectively). As a result, the fusion model further elevated the overall concern level (82.48%). According to the resource concern model, this output signifies a SEVERE concern level for the host in question. On the other hand, whenever there were disagreements between the results of incident and cluster associations, the combined overall concern decreased in accordance with the extent of the conflict between the two and undertook a compromised assessment (this happened for most of the hosts where there were conflicts between their reported IAS and CAS).

*R&A for RealSecure-MSU Sensor Report*

This experiment was conducted on the RealSecure-MSU sensor report. Table 4.29 shows the situation assessment for reported hosts with their overall degree of concern (ODOC) - determined by fusing their incident association strengths (IAS)s and cluster association strengths (CAS)s.

Table 4.29 Situation Assessment for the RealSecure-MSU Sensor Report

| Dataset | Host | IAS | CAS | ODOC |
|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 172.016.112.010 | 70.80 | 65.51 | 67.12 |
| | 172.016.112.050 | 70.80 | 65.51 | 67.12 |
| | 172.016.115.020 | 70.80 | 65.51 | 67.12 |
| LLDOS 1.0 DMZ | 172.016.115.020 | 34.00 | 80.86 | 55.65 |
| | 172.016.112.050 | 34.00 | 65.51 | 49.35 |
| | 172.016.114.010 | 34.00 | 65.51 | 49.35 |
| | 172.016.114.020 | 34.00 | 65.51 | 49.35 |
| | 172.016.114.030 | 34.00 | 65.51 | 49.35 |
| | 172.016.114.050 | 32.03 | 65.51 | 47.84 |
| | 172.016.114.001 | 32.03 | | 25.27 |
| | 172.016.112.010 | 14.00 | | 15.46 |
| LLDOS 2.0.2 Inside Zone | 172.016.112.050 | 48.70 | 65.51 | 56.79 |
| | 172.016.115.020 | 54.19 | 65.51 | 56.79 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| LLDOS 2.0.2 DMZ | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.115.020 | 18.67 | 69.00 | 39.40 |
| | 172.016.114.001 | 32.03 | | 25.27 |
| | 172.016.114.050 | 32.03 | | 25.27 |

Situation assessment with the RealSecure-MSU sensor report yielded good results. Table 4.29 shows that the only hosts with an ELEVATED level of concern were the actual victim hosts of the LLD attack experiments. A CAUTIOUS level of concern represents hosts, which were pursued in the attacks, except the host *robin*: 172.016.112.207. This host was not compromised but there were evidence of multiple

*finger* attacks present in the sensor report that resulted in a cluster generated for the host (explained in Experiment 4.3.3B). The following discusses interesting examples from Table 4.29:

In the case of LLDOS 1.0 DMZ dataset, for one of the victim hosts *locke*: 172.016.112.010, there was no report for cluster association, however incident association was reported as 14.0% (IAS). This resulted in an overall degree of concern of 15.46%. According to the resource concern model, this output signifies a LOW level of concern for the host. Figure 4.41 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

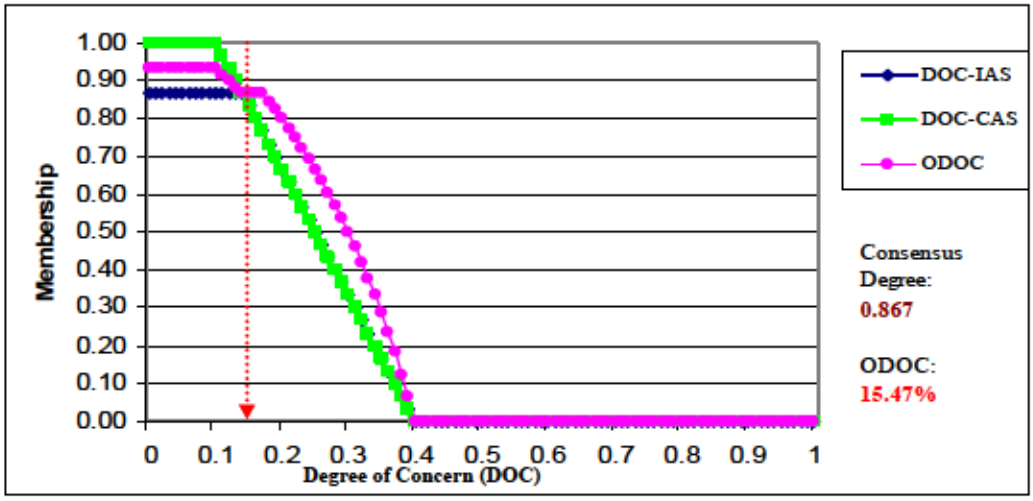


Figure 4.41 Dynamic Fusion Results for Host *locke* for LLDOS 1.0 DMZ Dataset analyzing the RealSecure-MSU Sensor Report

Both of the possibility distributions of degree of concern related by IAS (DOC-IAS) and degree of concern related by CAS (DOC-CAS) consist of *Low* fuzzy sets and show weak conflict between them. As expected, the combined output fuzzy set shows additive behavior within the level of consensus (0.867) and compromised behavior beyond it. The arrow points to the defuzzified output value in the *Low* region.

In the case of LLDOS 2.0.2 DMZ dataset, for one of the victim hosts *mill*: 172.016.115.020, the incident association was reported as 18.67% (IAS) and cluster association was reported as 69.0% (CAS). Combining them resulted in an overall degree of concern of 39.40%. According to the resource concern model, this output signifies a LOW level of concern for the host. Figure 4.41 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

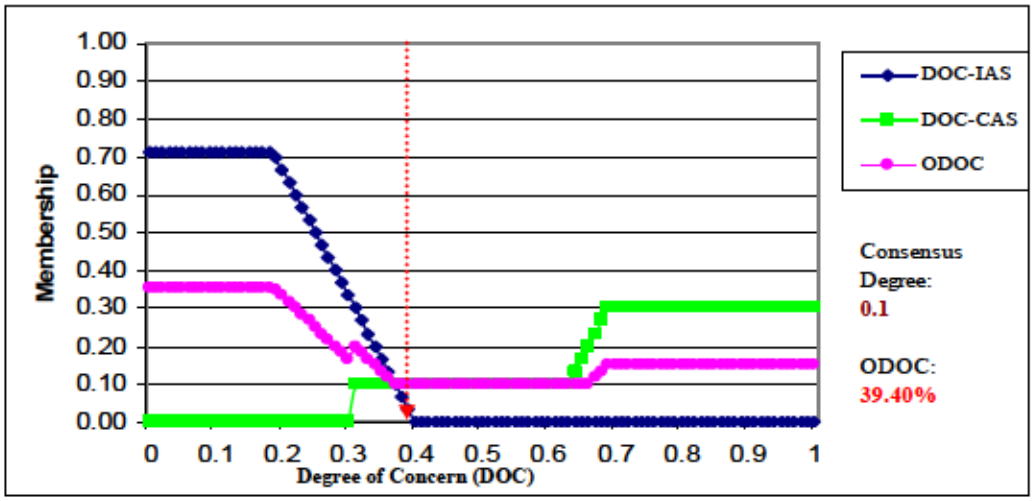


Figure 4.42 Dynamic Fusion Results for Host *mill* for LLDOS 2.0.2 DMZ Dataset analyzing the RealSecure-MSU Sensor Report

The possibility distribution of DOC-IAS consists of only a *Low* fuzzy set. However, possibility distribution of DOC-CAS has membership in both *Medium* and *High* regions to different extents - showing strong conflict with DOC-IAS. As expected, the combined output fuzzy set shows compromised behavior throughout the possibility distribution except at the level of consensus. The arrow points to the defuzzified output value at the lower end of the *Medium region*.

The results of situation assessment for this experiment also shows that whenever there was strong agreement between the results reported by abstract alert correlation and multi-level alert clustering, the combined overall concern increased (e.g., in the case of LLDOS 1.0 DMZ dataset and for the host *locke*: 172.016.112.010, IAS was reported as 14.0% and CAS was not reported (i.e., 0%). In possibility distribution, both of these resulted in *Low* fuzzy sets with little conflict. As a result, the overall concern was reported as 15.46%. According to the resource concern model, this output signifies a LOW concern level for the host in question. In contrast, whenever there were disagreements between the results of incident and cluster associations, the combined overall concern decreased in accordance with the extent of the conflict between the two and undertook a compromised assessment (this happened for most of the hosts where there were conflicts between their reported IAS and CAS).

*R&A for Snort-MSU Sensor Report*

This experiment was conducted on the Snort-MSU sensor report. Table 4.30 shows the situation assessment for reported hosts with their overall degree of concern (ODOC) - determined by fusing their incident association strengths (IAS)s and cluster association strengths (CAS)s.

Table 4.30 Situation Assessment for the Snort-MSU Sensor Report

| Dataset | Host | IAS | CAS | ODOC |
|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 172.016.112.050 | 100.00 | 88.42 | 84.72 |
| | 172.016.112.010 | 100.00 | 69.00 | 79.03 |
| | 172.016.115.020 | 100.00 | 69.00 | 79.03 |
| | 172.016.112.194 | 15.44 | 93.55 | 52.16 |
| | 172.016.114.050 | 34.37 | 65.51 | 49.84 |
| | 172.016.113.169 | | 87.54 | 49.14 |
| | 172.016.113.204 | | 87.54 | 49.14 |
| | 172.016.113.148 | | 80.86 | 45.77 |
| | 172.016.113.207 | | 77.43 | 43.32 |
| | 172.016.113.168 | | 75.94 | 42.07 |
| | 172.016.112.149 | | 65.51 | 40.41 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.084 | | 65.51 | 40.41 |
| | 172.016.113.105 | | 65.51 | 40.41 |
| | 172.016.112.100 | | 63.32 | 37.70 |
| LLDOS 1.0 DMZ | 172.016.112.050 | 60.00 | 80.86 | 64.04 |
| | 172.016.112.010 | 60.00 | 69.00 | 60.35 |
| | 172.016.114.010 | 60.00 | 69.00 | 60.35 |
| | 172.016.114.020 | 60.00 | 69.00 | 60.35 |
| | 172.016.114.030 | 60.00 | 69.00 | 60.35 |
| | 172.016.115.020 | 60.00 | 69.00 | 60.35 |
| | 172.016.114.050 | 34.37 | 83.95 | 57.57 |
| | 172.016.112.194 | 15.44 | 80.86 | 47.93 |
| | 172.016.112.149 | | 80.86 | 45.77 |
| | 172.016.113.084 | | 80.86 | 45.77 |
| | 172.016.113.148 | | 80.86 | 45.77 |
| | 172.016.113.168 | | 80.86 | 45.77 |
| | 172.016.113.169 | | 80.86 | 45.77 |
| | 172.016.113.204 | | 80.86 | 45.77 |
| | 172.016.113.207 | | 80.86 | 45.77 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.105 | | 65.51 | 40.41 |

Table 4.30 Situation Assessment for the Snort-MSU Sensor Report (continued)

| Dataset | Host | IAS | CAS | ODOC |
|---|---|---|---|---|
| LLDOS 2.0.2 Inside Zone | 172.016.115.020 | 72.80 | 80.86 | 83.15 |
| | 172.016.112.050 | 72.80 | 69.00 | 75.25 |
| | 172.016.112.100 | | 93.27 | 50.00 |
| | 172.016.113.148 | | 80.86 | 45.77 |
| | 172.016.113.168 | | 80.48 | 45.52 |
| | 172.016.113.050 | | 78.72 | 44.31 |
| | 172.016.112.149 | | 77.43 | 43.32 |
| | 172.016.112.194 | | 75.94 | 42.07 |
| | 172.016.113.204 | | 75.94 | 42.07 |
| | 172.016.113.169 | | 65.51 | 40.41 |
| | 172.016.113.084 | | 62.44 | 36.65 |
| LLDOS 2.0.2 DMZ | 172.016.115.020 | 34.37 | 69.00 | 52.21 |
| | 172.016.112.100 | | 88.92 | 49.65 |
| | 172.016.112.194 | | 80.86 | 45.77 |
| | 172.016.113.204 | | 80.86 | 45.77 |
| | 172.016.113.105 | | 65.51 | 40.41 |
| | 172.016.112.149 | | 69.00 | 36.10 |
| | 172.016.113.050 | | 69.00 | 36.10 |
| | 172.016.113.084 | | 69.00 | 36.10 |
| | 172.016.113.168 | | 69.00 | 36.10 |

Misuse situation assessment with the Snort-MSU sensor report yielded good results. Table 4.30 shows that all the hosts, which were under attack in the LLD attack experiments, were reported with a SEVERE, HIGH or ELEVATED level of concern, depending on the extent of attack evidence found in the sensor report. Two additional hosts were reported with an ELEVATED level of concern (*falcon:* 172.016.112.194 and *marx:* 172.016.114.050, for LLDOS 1.0 Inside Zone and DMZ datasets respectively). As explained in Experiments 4.3.2C and 4.3.3C, although these hosts were not under attack, the fusion model correlated and clustered alerts for these hosts because there was evidence of suspicious *Ping* and *Telnet* alerts in the sensor reports. In Table 4.30, a CAUTIOUS or a LOW level of concern represents hosts, which were not compromised but for which there were evidence of some form of anomalous activities present in the

sensor report that resulted in incidents activated or clusters generated or both. The following discusses interesting examples from Table 4.30:

In the case of LLDOS 1.0 DMZ dataset, for the DMZ host *plato*: 172.016.114.010, the incident association was reported as 60.0% (IAS) and cluster association was reported as 69.0% (CAS). Combining these resulted in an overall degree of concern of 60.35%. According to the resource concern model, this output signifies an ELEVETED level of concern for the host. Figure 4.43 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

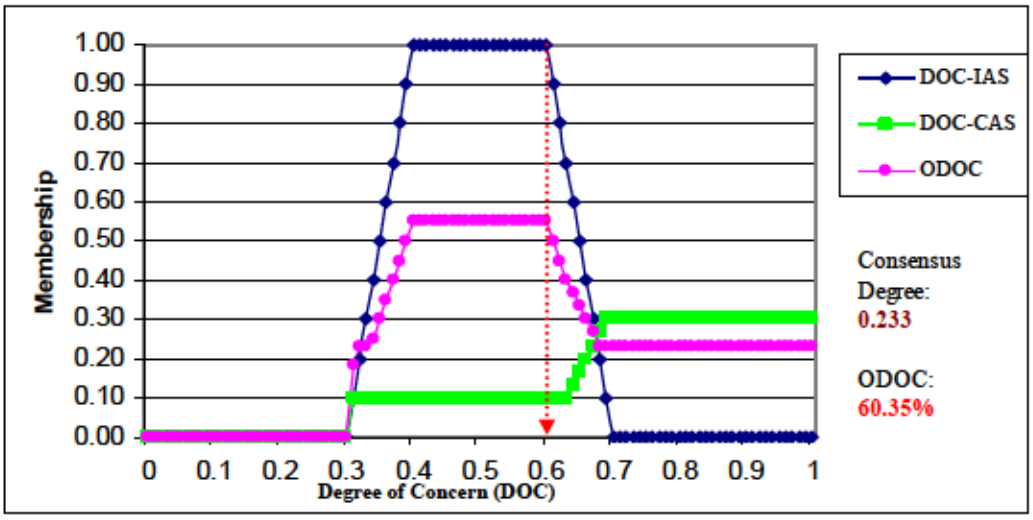


Figure 4.43 Dynamic Fusion Results for Host *plato* for LLDOS 1.0 DMZ Dataset analyzing the Snort-MSU Sensor Report

Both the possibility distributions of DOC-IAS and DOC-CAS consist of *Medium* fuzzy sets. In addition, DOC-CAS has membership in the *High* fuzzy set. As expected, the combined output fuzzy set shows compromised behavior beyond the level of

consensus (0.233). The arrow points to the defuzzified output value at the higher end of the *Medium* region.

In the case of LLDOS 2.0.2 Inside Zone dataset, for the victim host *pascal*: 172.016.112.050, the incident association was reported as 72.8% (IAS) and cluster association was reported as 69.0% (CAS). Combining these resulted in an overall degree of concern of 75.25%. According to the resource concern model, this output signifies a HIGH level of concern for the host. Figure 4.44 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

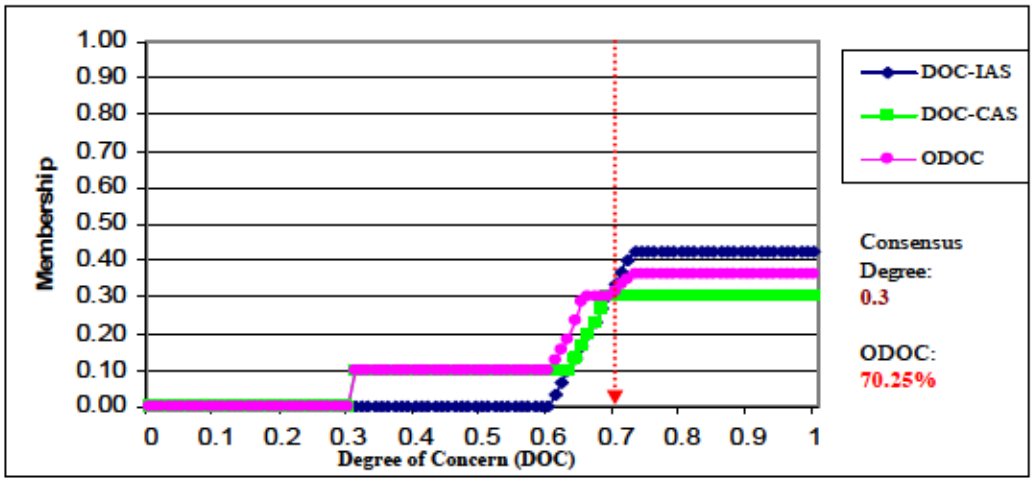


Figure 4.44 Dynamic Fusion Results for Host *pascal* for LLDOS 2.0.2 Inside Zone Dataset analyzing the Snort-M

The possibility distribution of DOC-IAS consists of only a *High* fuzzy set, while that of DOC-CAS has membership in both *Medium* and *High* fuzzy sets. As expected, the combined output fuzzy set shows additive behavior when the inputs agree. When there is conflict, it follows the level of consensus (0.30), unless any of the inputs passess this

level – in that case, compromised behavior follows. The arrow points to the defuzzified output value at the lower end of the *High* region.

The results of situation assessment for this experiment also shows that whenever there was strong agreement between the results reported by abstract alert correlation and multi-level alert clustering, the combined overall concern increased (e.g., in the case of LLDOS 2.0.2 Inside Zone dataset and for the host *pascal*: 172.016.112.050, similar values for IAS and CAS were reported (72.80% and 69.0% respectively). As a result, the fusion model further elevated the overall concern level (75.25%). According to the resource concern model, this output signifies a HIGH concern level for the host in question. Again, whenever there were disagreements between the results of incident and cluster association, the combined overall concern decreased in accordance with the extent of the conflict between the two and undertook a compromised assessment (this happened for most of the hosts where there were conflicts between their reported IAS and CAS).

*R&A for MultiSensor-MSU Report*

This experiment was conducted for the MultiSensor-MSU report. Table 4.31 shows the situation assessment for reported hosts with their overall degree of concern (ODOC) - determined by fusing their incident association strengths (IAS)s and cluster association strengths (CAS)s.

Table 4.31 Situation Assessment for MultiSensor-MSU Report

| Dataset | Host | IAS | CAS | ODOC |
|---|---|---|---|---|
| LLDOS 1.0 Inside Zone | 172.016.112.050 | 100.00 | 93.55 | 84.83 |
| | 172.016.112.010 | 100.00 | 88.92 | 84.76 |
| | 172.016.115.020 | 100.00 | 88.92 | 84.76 |
| | 172.016.112.194 | 15.00 | 93.55 | 51.95 |
| | 172.016.113.169 | | 88.42 | 49.47 |
| | 172.016.114.050 | 34.00 | 65.51 | 49.35 |
| | 172.016.113.204 | | 87.54 | 49.14 |
| | 172.016.113.168 | | 85.08 | 48.09 |
| | 172.016.113.148 | | 80.86 | 45.77 |
| | 172.016.113.207 | | 77.43 | 43.32 |
| | 172.016.112.100 | | 76.62 | 42.66 |
| | 172.016.112.149 | | 65.51 | 40.41 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.084 | | 65.51 | 40.41 |
| | 172.016.113.105 | | 65.51 | 40.41 |
| LLDOS 1.0 DMZ | 172.016.114.050 | 73.00 | 91.52 | 84.18 |
| | 172.016.112.050 | 60.00 | 88.92 | 66.17 |
| | 172.016.114.010 | 60.00 | 88.92 | 66.17 |
| | 172.016.114.020 | 60.00 | 88.92 | 66.17 |
| | 172.016.114.030 | 60.00 | 88.92 | 66.17 |
| | 172.016.115.020 | 60.00 | 88.92 | 66.17 |
| | 172.016.112.010 | 60.00 | 80.86 | 64.04 |
| | 172.016.112.194 | 15.00 | 80.86 | 47.71 |
| | 172.016.112.149 | | 80.86 | 45.77 |
| | 172.016.113.084 | | 80.86 | 45.77 |
| | 172.016.113.148 | | 80.86 | 45.77 |
| | 172.016.113.168 | | 80.86 | 45.77 |
| | 172.016.113.169 | | 80.86 | 45.77 |
| | 172.016.113.204 | | 80.86 | 45.77 |
| | 172.016.113.207 | | 80.86 | 45.77 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.105 | | 65.51 | 40.41 |
| | 172.016.114.001 | 32.00 | | 25.17 |

Table 4.31 Situation Assessment for MultiSensor-MSU Report (continued)

| Dataset | Host | IAS | CAS | ODOC |
|---|---|---|---|---|
| LLDOS 2.0.2 Inside Zone | 172.016.112.050 | 73.00 | 88.92 | 84.08 |
| | 172.016.115.020 | 79.00 | 80.86 | 83.26 |
| | 172.016.112.100 | | 93.27 | 50.00 |
| | 172.016.112.194 | | 85.08 | 48.09 |
| | 172.016.113.148 | | 80.86 | 45.77 |
| | 172.016.113.168 | | 80.48 | 45.52 |
| | 172.016.113.050 | | 80.08 | 45.13 |
| | 172.016.112.149 | | 77.43 | 43.32 |
| | 172.016.113.204 | | 77.43 | 43.32 |
| | 172.016.113.084 | | 75.02 | 41.24 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.169 | | 65.51 | 40.41 |
| LLDOS 2.0.2 DMZ | 172.016.115.020 | 41.00 | 69.00 | 60.35 |
| | 172.016.112.100 | | 88.92 | 49.65 |
| | 172.016.112.194 | | 80.86 | 45.77 |
| | 172.016.113.204 | | 80.86 | 45.77 |
| | 172.016.112.207 | | 65.51 | 40.41 |
| | 172.016.113.105 | | 65.51 | 40.41 |
| | 172.016.112.149 | | 69.00 | 36.10 |
| | 172.016.113.050 | | 69.00 | 36.10 |
| | 172.016.113.084 | | 69.00 | 36.10 |
| | 172.016.113.168 | | 69.00 | 36.10 |
| | 172.016.114.001 | 32.00 | | 25.17 |
| | 172.016.114.050 | 32.00 | | 25.17 |

Misuse situation assessment with the MultiSensor-MSU report yielded good results. Table 4.31 shows that all the hosts, which were under attack in the LLD attack experiments, were reported with a SEVERE or ELEVATED level of concern, depending on the extent of attack evidence found in the sensor report. Two additional hosts were reported with similar levels of concern (*falcon:* 172.016.112.194 and *marx:* 172.016.114.050, for LLDOS 1.0 Inside Zone and DMZ datasets respectively). As explained in Experiments 4.3.2D and 4.3.3D, although these hosts were not under attack, the fusion model correlated and clustered alerts for these hosts because there was evidence of suspicious alerts in the sensor reports.

In Table 4.31, CAUTIOUS or LOW levels of concern represent hosts, which were not compromised but for which there were evidence of some form of anomalous activities present in the sensor report that resulted in incidents activated or clusters generated or both. The following discusses interesting examples from Table 4.31:

In the case of LLDOS 1.0 Inside Zone dataset, for the inside host *crow*: 172.016.113.148, there was no report of incident association and cluster association was reported as 80.86% (CAS). This resulted in an overall degree of concern of 45.78%. According to the resource concern model, this output signifies a CAUTIOUS level of concern for the host. Figure 4.45 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

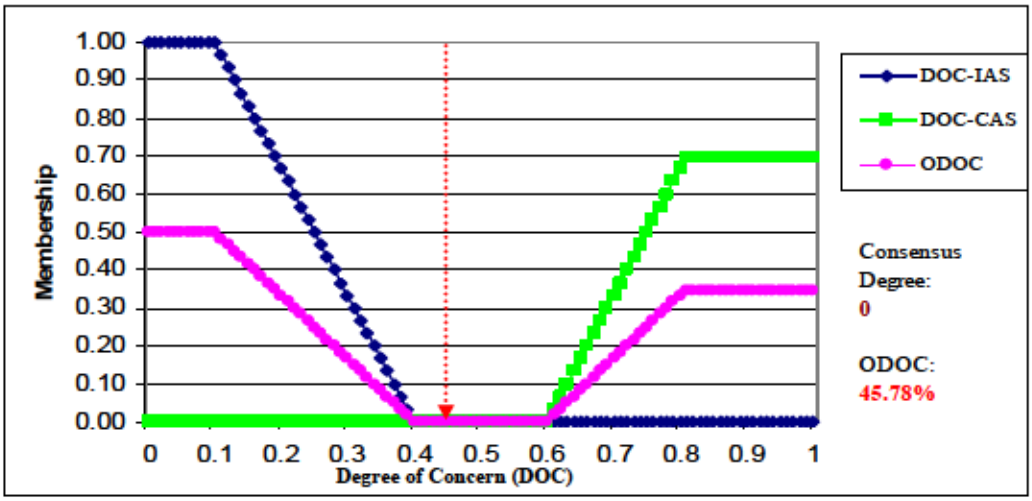


Figure 4.45 Dynamic Fusion Results for Host *crow* for LLDOS 1.0 Inside Zone Dataset analyzing the MultiSensor-MSU Report

The possibility distribution of DOC-IAS consists of only a *Low* fuzzy set, while that of DOC-CAS consists of only a *High* fuzzy set, showing total conflict between the two. As expected, the combined output fuzzy set shows compromised behavior throughout because of the zero consensus degree. The arrow points to the defuzzified output value at the lower end of the *Medium* region.

In the case of LLDOS 2.0.2 Inside Zone dataset, for the victim host *mill*: 172.016.115.020, incident association was reported as 80.86% (IAS) and cluster association was reported as 79.0% (CAS). Combining these resulted in an overall degree of concern of 83.26%. According to the resource concern model, this output signifies a SEVERE level of concern for the host. Figure 4.46 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

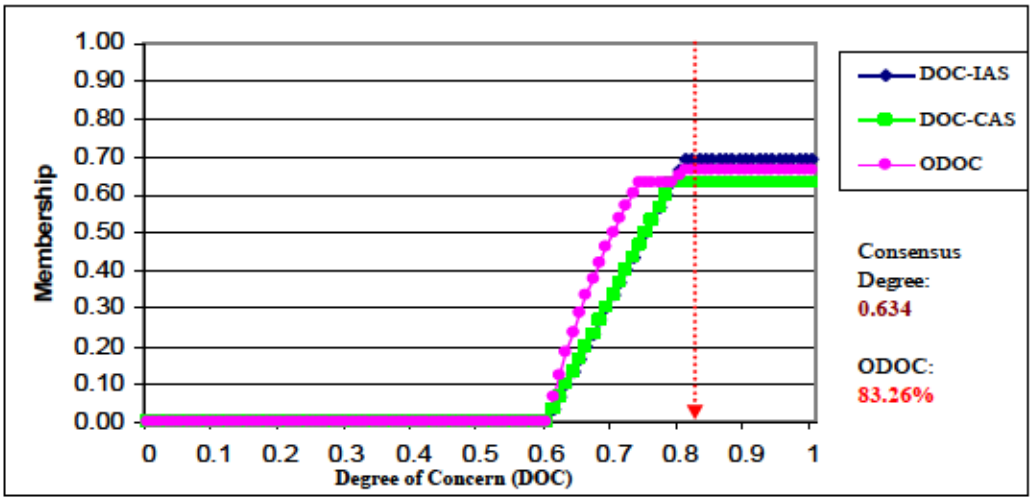


Figure 4.46 Dynamic Fusion Results for Host *mill* for LLDOS 2.0.2 Inside Zone Dataset analyzing the MultiSensor-MSU Report

Both of the possibility distributions of degree of concern related by IAS (DOC-IAS) and degree of concern related by CAS (DOC-CAS) consist of *High* fuzzy sets and show weak conflict between them. As expected, the combined output fuzzy set shows additive behavior until the level of consensus (0.633) is passed and conflict occurs. After that output fuzzy set follows compromised behavior. The arrow points to the defuzzified output value in the *High* region.

The results of situation assessment for this experiment confirms that whenever there was strong agreement between the results reported by abstract alert correlation and multi-level alert clustering, the combined overall concern increased (e.g., in the case of LLDOS 2.0.2 Inside Zone dataset and for the host *mill*: 172.016.115.020, as explained above).

In contrast, whenever there were disagreements between the results of the incident and cluster association, the combined overall concern decreased in accordance to the extent of the conflict between the two and undertook a compromised assessment (this happened for most of the hosts where there were conflicts between their reported IAS and CAS).

*Summary for Misuse Situation Assessment Experiment*

In all of the experiments conducted for misuse situation assessment, we found that the fusion model was able to combine the results of abstract alert correlation and multi-level alert clustering to provide quantitative assessments that represent the overall degree of concern for the designated hosts in their involvement in security situations (i.e., situations involving multi-staged attacks and common attack patterns). The dynamic fusion technique for misuse situation assessment used agreement between the results of

abstract alert correlation and multi-level alert clustering to guide the fusion process. We found that beyond the level of consensus when the two results disagreed, compromised behavior took place, and within the level of consensus, when the results agreed, additive behavior took place and when they did not, the level of consensus was followed. Thus the fused result for final situation assessment was indicative of the collective extent of concern generated from both alert correlation and alert clustering.

### *4.3.5 Anomaly Situation Assessment Experiment*

*Objective*

Anomaly situation assessment with sensor corroboration provides an overall alert assessment for hosts for which anomaly sensors report alerts. The final assessment combines event-based evidence by primary sensor in the form of an event anomaly with that reported by the secondary sensor monitoring changes in system state attributes. This experiment is designed to evaluate the dynamic fusion process for anomaly situation assessment.

*Results and Analysis (R&A)*

The following are the results and analysis of the situation assessment experiment with synthetic data representing event- and state-based evidence reported by a primary sensor and a secondary sensor. It should be noted that anomaly situation assessment takes place only when the anomaly sensor reports anomalies for monitored hosts.

Table 4.32 shows the situation assessment results for certain hosts with their overall degree of concern (ODOC) that was determined by fusing event anomalies reported by the primary sensor and hosts' system state attribute alterations reported by the secondary sensor (in this case, we used change in available memory as the state-based evidence).

Table 4.32 Situation Assessment for Sensor Corroboration

| Host | Event Anomaly % Reported by Primary Sensor | System State Attribute (Available Memory) Alteration % Reported by Secondary Sensor | ODOC |
|---|---|---|---|
| Host 1 | 80.00 | 10.00 | 63.78 |
| Host 2 | 10.00 | 90.00 | 31.86 |
| Host 3 | | 100.00 | 31.86 |
| Host 4 | 100.00 | | 68.14 |
| Host 5 | 25.00 | 75.00 | 33.14 |
| Host 6 | 75.00 | 25.00 | 66.86 |
| Host 7 | 91.78 | 72.88 | 85.80 |
| Host 8 | | 65.51 | 35.39 |
| Host 9 | 14.00 | | 14.13 |
| Host 10 | 18.67 | 69.00 | 30.36 |
| Host 11 | 72.80 | 69.00 | 76.21 |
| Host 12 | 10.00 | 10.00 | 10.00 |
| Host 13 | 100.00 | 100.00 | 100.00 |

Anomaly situation assessment for sensor corroboration yielded good results. Table 4.32 shows that the hosts reported with SEVERE, HIGH or ELEVATED levels of concerns had more anomaly (>70%) reported by the primary sensor. In Table 4.31, LOW levels of concern represent hosts for which reports of anomaly were not as much, although, in some cases, secondary sensor reported noticeable change in available memory. It should be noted that when both reports were the same (last two row), the

overall degree of concern remained the same. The following discusses interesting examples from Table 4.32:

For host 11, the primary sensor reported a 72.8% event anomaly and the secondary sensor reported a 69.0% change in available memory. This resulted in an overall degree of concern of 76.21%. According to the resource concern model, this output signifies an HIGH level of concern for the host. This is justifiable since both sensors reported very similar evidence support. Figure 4.47 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

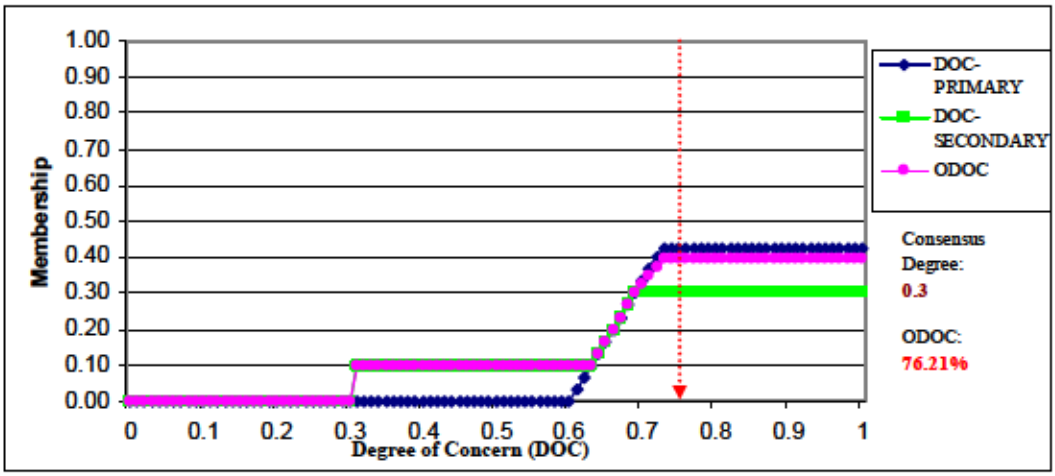


Figure 4.47 Dynamic Fusion Results for Host 11

The possibility distributions of degree of concern related by event anomaly reported by the primary sensor (DOC-PRIMARY) and degree of concern related by system state attribute alteration reported by the secondary sensor (DOC-SECONDARY) both consist of *High* fuzzy sets. As expected, the combined output fuzzy set shows disjunctive behavior until the level of consensus, beyond which compromised behavior with bias

towards the primary sensor report takes place. The arrow points to the defuzzified output value at the lower end of the *High* region.

For host 5, the primary sensor reported a 25% event anomaly and the secondary sensor reported a 75% change in available memory. This resulted in an overall degree of concern of 33.14%. According to the resource concern model, this output signifies a LOW level of concern for the host. This is justifiable because although the secondary sensor reported substantial change in available memory (which may happen for non-malicious/legitimate program execution), however the primary sensor report was still minor. Figure 4.48 shows the possibility distribution of the inputs and the output of the dynamic fusion process in this case.

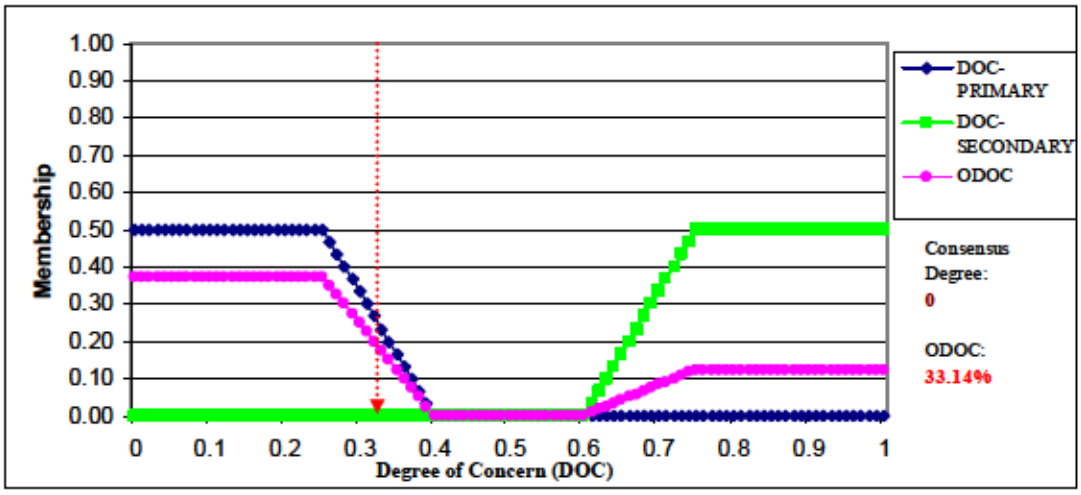


Figure 4.48 Dynamic Fusion Results for Host 5

The possibility distributions of DOC-PRIMARY consist of a *Low* fuzzy set, while that of DOC-SECONDARY consists of a *High* fuzzy set. As expected, the combined output fuzzy set shows compromised behavior with bias towards the primary sensor report. The arrow points to the defuzzified output value at the higher end of the *Low* region.

*Summary for Anomaly Situation Assessment Experiment*

In the experiment conducted for anomaly situation assessment, we found that the fusion model was able to combine the reports by the primary and the secondary sensors to provide quantitative assessments that represent overall degree of concerns for designated hosts in their involvement in anomalous security situations. The dynamic fusion technique for anomaly situation assessment used agreement between the reports to guide the fusion process. We found that beyond the level of consensus when the two results disagreed, compromised behavior took place with bias towards the results of the primary sensor, and within the level of consensus, when the results agreed, disjunctive behavior took place and when they did not, the level of consensus was followed. Thus the fused result for final situation assessment was indicative of the collective extent of concern generated from both primary and secondary sensors with bias towards the primary sensor's report.

## 4.4 Summary of Results

The main objective of the experiments was to investigate if the unified alert fusion model developed as part of this research was able to conduct high level reasoning of the low level sensor reported data in such a way that provided the security administrator with a condensed view of systems' security health. Experiments were conducted for each of the primary sensor fusion task of the unified alert fusion model (i.e., alert prioritization, alert correlation and alert clustering) and then the individual results were assessed for evaluating the performance of each of the sensor fusion tasks and their overall effect on the final situation assessment results for the hosts.

The alert prioritization experiment demonstrated that the unified alert fusion model reduced alert volume to a great extent by identifying low priority alerts such that the low priority alerts could be excluded from further analysis. Such alert reduction proved to be useful in terms of saving processing time considerably. When we conducted alert correlation and alert clustering without using the results of alert prioritization, i.e., without any filtering, the fusion model had to process 2737, 246, 7467, 7713 alerts for the RealSecure-NCSU, RealSecure-MSU, Snort-MSU and MultiSensor-MSU report respectively. However, with filtering, the fusion model only had to analyze 282, 108, 437, 545 alerts for the RealSecure-NCSU, RealSecure-MSU, Snort-MSU and MultiSensor-MSU report respectively. Furthermore, we observed that the filtering of the low priority alerts did not affect the situation assessment results for victim hosts for any of the sensor reports.

In the alert correlation experiment the fusion model successfully reported all the victim hosts of the LLD attack experiment. In some cases, a few additional hosts were also reported under attack with low incident association strengths, when the fusion model incorrectly correlated alerts for them. This happened in situations when isolated alerts contribute to chronological incidents in correlation chain. Also, the fusion model missed a few alerts in situations when communication occurred between IP addresses outside of the resource perimeter, when alerts would generalize to certain abstract types that would not contribute to any coordinated attack scenario, or when correlated alerts contributed to out of sequence incidents in the correlation chain. Our result for alert correlation was found to be comparable with results reported by NCSU on the same sensor report [37]. This is encouraging because ours is a simpler technique and the scenario model is more generic in the sense that variations in data are more easily accommodated without changing the model or having to codify specific knowledge about individual intrusions/alerts. The result obtained with multi-sensor report demonstrated the potential of the fusion model to link together seemingly isolated incidents inferred from evidence reported by different sensors to discover coordinated attack scenarios. Result with missing alerts in sensor report showed the capability of the fusion model to continue its reasoning process even with missing evidence of attack, considering only the situation leading up to the incidents. Also, we found that alert fusion with the abstract incident model aided in alert reduction by reporting only correlated alerts. Incident association provided the security administrator with a quick insight into the incident situation for hosts involved in multi-staged attacks.

In the alert clustering experiment, the fusion model successfully reported all the victim hosts of the LLD attack experiment. In some cases, additional hosts were also reported when alert clusters with common features were found for them. This happened because multi-level clustering does not analyze the significance of alerts but only the occurrence of alerts with similarity between them. Result obtained with multi-sensor report demonstrated the potential of the fusion model to associate seemingly dispersed alerts with similar features to discover common attack patterns within multiple sensor reports. We also found that alert fusion with multi-level clustering aided in alert reduction by reporting alert clusters. Cluster association provided the security administrator with a quick insight into the cluster situation for hosts involved in common attack patterns.

Final situation assessment experiment demonstrated how the results of alert correlation and alert clustering were fused to provide condensed views of the security health of hosts in the protected network in terms of reporting overall degree of concerns for their involvement in security situations. Using a resource concern model, the extent of concerns for reported hosts were conveyed to the security administrator in such a way that allowed prompt identification of the hosts that were under attack (hosts with a Severe, High or Elevated concern level) from those that were not but for which there were reasons for some concerns (hosts with a Cautious or Low concern level). Intuitively such a view that provide insight into the nature and severity of security incidents and common attack patterns are much more informative and intelligible to the security administrator than simply a log of large volume of alert records that the low level sensors generate.

# CHAPTER V

# CONCLUSIONS AND FUTURE WORK

This chapter summarizes the contributions of the unified alert fusion model approach to intelligent sensor fusion and the results of the experiments conducted as part of the research. The limitations of our current implementation are described. Lastly, we conclude with direction for potential future extensions of this research.

## 5.1 Contributions and Summary

This dissertation makes several contributions to the state of the art of sensor alert fusion in an intrusion detection environment. The key contribution is the development of a unified architecture for intelligent alert fusion that combines the primary tasks of sensor fusion in a single framework for overall security situation assessment of protected network resources in a distributed environment. Specific contributions in this regard are as follows:

In this dissertation, a new alert prioritization technique has been developed to filter lower priority alerts such that further analysis can focus on higher priority alerts. The alert prioritization process primarily takes into account three factors (source/target criticality, attack criticality and alert confidence) to assess the relative importance of the sensor alerts.

This research introduces a new alert clustering technique using fuzzy cognitive modeling with generalization to group/cluster alerts with the same and similar features in order to identify common attack patterns. The proposed multi-level alert clustering approach clusters alerts at different levels of abstraction or resolution such that different degrees of deviations in commonality of alert features are tolerated. In this way, along with identical alert clusters, clusters of "similar" alerts are also found.

In this research, a new alert correlation technique has been described that uses fuzzy cognitive modeling with generalization to correlate alerts that are linked in multi-staged attacks. We have developed an abstract incident model for alert correlation with generalized security events to deal with scalability issues in sensor fusion. By focusing on the effects of the intrusions, such an abstract incident model captures the essence of typical or commonly occurring techniques used the attackers in multi-staged attacks and correlates alerts, even though intermediate alerts are missing in the sensor reports.

This dissertation has defined a new concept of situation assessment that derives quantitative assessment of a systems' security health using a possibilistic approach. The family of dynamic fusion approaches introduced for situation assessment combines the

results of alert clustering and alert correlation in the case of misuse or signature-based sensors and combines the reports of event-based evidence and state-based evidence in the case of anomaly or profile-based sensors. The sensor corroboration concept for use with anomaly sensors can be particularly suitable for a resource restrained high performance cluster environment.

As part of this research, a new taxonomy for categorizing attacks has also been developed that is based on the possible impacts of probable attacks. The taxonomy would be useful for researchers or practitioners in the area of intrusion detection for broader understanding of attacks and their impacts.

In this dissertation, we have shown empirically that our hypothesis is valid. We present the first reported detailed empirical evaluation of multiple sensor fusion tasks conducted on multiple independent and integrated sensor alert reports generated on a well known benchmark attack dataset (i.e., MIT Lincoln Lab's DARPA 2000 Intrusion Detection Evaluation Scenario Specific dataset). The experiments conducted on individual and integrated sensor reports verify that the unified alert fusion model consistently performed well in combining the tasks of alert prioritization, alert clustering and alert correlation into a single framework for overall assessment of the systems' security health.

The main advantage of our alert prioritization technique has been shown to reduce alert volume drastically by filtering lower priority alerts. The multi-level clustering technique proved beneficial by finding clusters of similar attack patterns when there were

variations in data which would cause failure of traditional clustering to find clusters of the same attack patterns. Our clustering technique has been shown to appropriately derive quantitative assessments of protected resources' involvement in common attack patterns. The level of such involvement provided insight into the extent of attacks targeted towards a resource. In addition, the clustering technique has been shown to further reduce alert volume by reporting only clustered alerts. The main advantage of our alert correlation technique with abstract incident modeling has been shown to link together alerts that are involved in multi-staged coordinated attacks by considering both evidence of attacks present in the sensor reports and the possible occurrence of such attacks. The abstract incident model allowed inference to progress even though evidence of attacks was missing in the sensor reports. Our correlation technique has been shown to properly derive quantitative assessments of the protected resources' involvement in multi-staged attacks. The level of such involvement provided insight into the criticality of coordinated attacks targeted towards a resource. In addition, the correlation technique has been shown to further reduce alert volume by reporting only correlated alerts. Finally, situation assessment with dynamic fusion has been shown to effectively combine the results of alert clustering and alert correlation for deriving overall degree of concerns for the protected resources' involvement in security situations. Such a high-level condensed view lends instant insight into and a better understanding of the systems' security health than does low-level sensor reports with no intelligent analysis present. For use with anomaly sensors, dynamic fusion with sensor corroboration has been shown to effectively

substantiate event-based evidence reported by a primary sensor with state-based evidence reported by a secondary sensor for deriving the overall degree of concerns for the protected resources' involvement in anomalous situations. The proposed dynamic fusion approaches are particularly suited for applications where dynamically available information from diverse and disparate sources must be combined.

In this research, we have addressed a timely and significant research problem with a promising new approach. Unlike previous efforts in this area, for intelligent alert fusion, we have used a possibilistic approach with new application of the soft computing tool, Fuzzy Cognitive Maps (FCMs). Using FCMs for cognitive modeling is primarily attractive for the following reasons:

- FCMs offer a practical yet natural knowledge acquisition scheme that represents expert's knowledge in a structured way and works well with human expert's thinking.
- FCMs are particularly suitable in a dynamic environment such as network security because they are flexible enough to capture the adaptive nature of human knowledge and therefore one can easily add new concepts or delete idle/obsolete concepts as necessary without difficulty.
- FCMs are suitable for soft knowledge domains such as intrusion detection where systems concepts/ relationships and also the meta-system language are essentially fuzzy [28].
- FCMs help prevent certain kinds of knowledge extraction problems often encountered in traditional rule-based systems [6].

## 5.2 Limitations and Future Works

There are several limitations of this research that lend scope for additional investigation or improvement.

In this dissertation, we have used cognitive models with FCMs whose structures have been defined by human experts. However, the models are intuitive and generic and require little or no specialized knowledge. In the future, we will explore the use of adaptive FCMs, where the FCMs can self-learn and self-train like neural networks with minimal involvement of the human expert.

A critical assumption in this research is that meaningful generalization hierarchies have been defined for the alert features and that the sensor reported attacks are appropriately categorized into the developed attack generalization hierarchy or taxonomy. Defining such generalization hierarchies is a knowledge engineering task that has no single best way to be done. The generalization hierarchies used in this research are simply shown as examples to demonstrate the usefulness of our model.

In this research, we have conducted explicit alert correlation where we have used a predefined abstract incident model to correlate alerts. One possible extension can be to conduct implicit correlation where we can use historical data to build the correlation models.

Like any sensor fusion approaches, our intelligent analysis of sensor alerts largely depends on the underlying low-level sensors or intrusion detection systems to report the alerts. Although ours tolerates missing alerts to some degree, in the extreme case, if the sensors miss all attacks, our approach will not be successful.

Another assumption made by our alert correlation technique is that earlier attacks prepare for later attacks. Sometimes in reality, an attacker does not have to perform early attacks to prepare for later attacks and may launch a malicious attack without any earlier evidence of such attack. In these cases, our alert correlation technique would not report the alerts unless the attack is of the utmost criticality (i.e., of system distress type).

The main purpose of our alert prioritization technique is to filter lower priority alerts from higher priority alerts such that further analysis is not distracted by false positives or non-malicious data. Nonetheless, there are provisions for risks associated with such filtering in cases when carefully crafted less obvious attacks are appraised as a lower priority and ignored from analysis.

Although the resource centric view adopted by our fusion model helps to reduce alert volume by concentrating on alerts involving only the protected resources, sometimes such a view can ignore the evidence of obvious malicious attacks involving spoofed and external addresses.

Our current implementation of the unified alert fusion model makes use of an SQL-based database as a sensor alert repository in order to take advantage of the functionalities of a relational Database Management Systems (DBMS). Although any

DBMS-based application provides enormous convenience and support, it may suffer from performance penalty when a large amount of data is considered. Also, frequent interactions with a DBMS may lead to performance bottlenecks. One solution would be to minimize the interaction with the DBMS and use the DBMS for storage only.

A number of extensions are possible to the research conducted as part of this dissertation.

Currently, the unified alert fusion model uses a centralized scheme where alerts generated for multiple resources and from multiple sensors are integrated at a centralized data repository and all analysis is conducted by a central fusion unit. Such a scheme has limitations in terms of scalability. Also, a failure of the central unit can blind the fusion process. One potential extension would be to decentralize some of the sensor fusion tasks at local fusion units such that the responsibility of sensor fusion is delegated. For example, our resource centric fusion model can easily be adapted to delegate alert prioritization and alert clustering tasks at the resource level.

Another natural extension is the development of a user friendly GUI or visualization component with roll-up/drill-down capability. Thus, when security administrators are presented with the high-level condensed view of the protected system's health, they can use the GUI to gradually investigate the causes of system concerns, such as, clusters found, and/or incidents activated and relate them to the contributing specific details of evidence in the sensor reports.

The dynamic fusion approach described in this dissertation is not limited to fusing information in an intrusion detection environment. This approach has the potential to be applied in suitable problems in other application domains. Also, for this work, the dynamic fusion approach combines information from symmetric sources (for example, incident and cluster association strengths, state and event-based evidence). However, in the real world sources can be asymmetric. Therefore, a possible future extension can be to develop dynamic fusion for asymmetric sources. Moreover, in this dissertation, we allow fusion of only two inputs. Another research possibility lies in extending the dynamic fusion approach to incorporate more than two inputs.

Future work also lies in extending the abstract incident model developed in this dissertation such that all abstract categories in the attack generalization hierarchy are utilized. Currently, the abstract incident model uses a subset of the abstract categories that are deemed critical for multi-staged attacks.

In this research, we have focused on what has happened to a protected resource from evidence provided by sensor reports. In the future, we want to extend this work to predict an attacker's future plans such that we are able to report what might or is about to happen to a protected resource. This has the potential to warn the security administrator in advance and aid in preventing such attacks.

Another issue that is worth future investigation is the collaboration between multiple information sources to provide a more holistic view of security situations. Data from vulnerability scanners, honey pots, and performance monitoring systems can be utilized

in this respect. Also, we would like to investigate incorporation of dynamic generalization hierarchy for alert feature abstraction.

It should be noted that the DARPA data is not intended to be conclusive examination of the effectiveness of our approach, but rather to provide a sense of how well and how accurate our approach works. Since our model has not been tested on a live system, a potential future research effort will be to experiment with real-time traffic in both distributed and cluster environment and with larger datasets.

Besides refining and extending this research, we intend to further study possible ways to integrate our approach with other complementary research in this area.

## 5.3 Related Publications

A list of publications related to this work is presented below:

- S. M Bridges, R. B. Vaughn, and A. Siraj. "AI Techniques Applied to High Performance Computing Intrusion Detection" *Proceeding: 10th International Conference on Telecommunication Systems, Modeling and Analysis*, Monterey, CA, vol. 2, Oct. 2002, pp. 100-114.

- A Siraj, S. M. Bridges, and R. B. Vaughn, "Fuzzy Cognitive Maps for Decision Support in an Intelligent Intrusion Detection System," *Proceedings: International Fuzzy Systems Association/ North American Fuzzy Information Processing Society (IFSA/NAFIPS) Conference on Soft Computing,* Vancouver, Canada, Jul. 2001.

- A. Siraj, R. B. Vaughn and S. M. Bridges, "Intrusion Sensor Data Fusion in an Intelligent Intrusion Detection System Architecture." *Proceedings: Frameworks and Methods for the Study and Analysis of Trust in Information Systems: Minitrack in the Software Technology Track of the Thirty-Seventh Hawaii International Conference on System Sciences (HICSS-37)*, Hawaii, Jan. 2004.

- A. Siraj, R. B. Vaughn, and S. M. Bridges, "Decision Making for Network Health Assessment in an Intelligent Intrusion Detection System Architecture," *International Journal of Information Technology and Decision Making,* vol. 3, no. 2, 2004.

- A. Siraj, and R. B. Vaughn, "Multi-Level Alert Clustering for Intrusion Detection Sensor Data", Proceedings: *North American Fuzzy Information Processing Society International Conference on Soft Computing for Real World Applications*, held in Ann Arbor, Michigan, June 2005.

- A. Siraj, and R. B. Vaughn, "A Cognitive Model for Alert Correlation in a Distributed Environment", Proceedings: *IEEE International Conference on Intelligence and Security Informatics (ISI 2005)*, Lecture Notes in Computer Science, Springer-Verlag, Volume 3495/2005.

BIBLIOGRAPHY


[1] D. Andersson, M. Fong, and A. Valdes, "Heterogeneous Sensor Correlation: A Case Study of Live Traffic Analysis," *Proceedings: IEEE Information Assurance Workshop*, United States Military Academy, West Point, NY, Jun. 2002.

[2] R. Axelrod, *Structured Decision: The Cognitive Maps of Political Elites*, Princeton University Press, Princeton, NJ, 1976.

[3] I. Bloch, "Information Combination Operators for Data Fusion: A Comparative Review with Classification", *IEEE Transactions on System, Man, and Cybernetics – Part A: Systems and Humans*, vol. 26, no. 1, pp. 52-67.

[4] S. M Bridges, R. B. Vaughn, and A. Siraj. "AI Techniques Applied to High Performance Computing Intrusion Detection" *Proceeding: 10th International Conference on Telecommunication Systems, Modeling and Analysis*, Monterey, CA, Oct. 2002, vol. 2, pp. 100-114.

[5] D. Brubaker, "Fuzzy Cognitive Maps," *EDN Access*, Apr. 1996.

[6] M. Caudill, "Using Neural Nets: Fuzzy Cognitive Maps", *AI Expert*, vol. 6, 1990, pp. 49-53.

[7] F. Cuppens, "Managing Alerts in a Multi-Intrusion Detection Environment," *Proceedings: 17th Annual Computer Security Applications Conference*, New Orleans, LA, Dec. 2001.

[8] F. Cuppens and A. Miege, "Alert Correlation in a Cooperative Intrusion Detection Framework," *Proceedings: 17th Annual Computer Security Applications Conference*, New Orleans, LA, Dec. 2001.

[9] O. M. Dain and R. K. Cunningham, "Building Scenarios from a Heterogeneous Alert Stream," *IEEE Transactions on Systems, Man and Cybernetics*, 2002.

[10] H. Debar and A. Wespi, "Aggregation and Correlation of Intrusion-Detection Alerts," *Proceedings: 4th International Symposium on Recent Advances in Intrusion Detection* (*RAID'04*), Davis, CA, Oct. 2001.

[11] V. Dhar and R. Stein, *Seven Methods for Transforming Corporate Data into Business Intelligence,* Prentice Hall Inc., NJ, 1997.

[12] D. Dubois, H. Prade, "Combination of information in the framework of possibility theory", *Data Fusion in Robotics and Machine Intelligence,* M. Al Abidi et al., ed., Academic Press, New York, NY, 1992.

[13] G. Florez, Z. Liu, S. M. Bridges, A. Skjellum, and R. B. Vaughn, "Lightweight monitoring of MPI programs in real-time," *Concurrency and Computation: Practice and Experience*, Nov. 2005, vol. 17, no. 3, pp. 1547-1578.

[14] H. Gao, *A Presentation Tool for the Intelligent Intrusion Detection System*, masters project report, Department of Computer Science and Engineering, Mississippi State University, 2002.

[15] R. P. Goldman, W. L. Heimerdinger, S. A. Harp, C. Geib, V. Thomas, R. L Carter, "Information Modeling for Intrusion Report Aggregation," *Proceedings: DARPA Information Survivability Conference and Exposition*, June 2001, *IEEE Computer Society*, pp. 329-342.

[16] J. Haines, D. K. Ryder, L. Tinnel and S. Taylor, "Validation of Sensor Alert Correlators," *IEEE Security and Privacy*, Jan./Feb. 2003.

[17] H. J., Hamilton, R. J. Hilderman, and N. Cercone, "Attribute-Oriented Induction Using Domain Generalization Charts", *Proceedings: 8th International Conference on Tools with Artificial Intelligence (ICTAI '96)*, Nov. 1996, pp.246.

[18] J. Howard and T. Longstaff, *A Common Language for Computer Security Incidents*, technical report SAND98-8667, Livermore, CA, Sandia National Laboratories, 1998.

[19] Imperial College Department of Computing, "Free Online Dictionary of Computing," http://wombat.doc.ic.ac.uk/foldoc/index.html (current Jun. 24, 2004)

[20] Internet Engineering Task Force (IETF), "Intrusion Detection Exchange Format (idwg)," http://www.ietf.org/html.charters/idwg-charter.html (current Sep. 29, 2003)

[21] Internet Security Systems, "RealSecure Network 10/100," http://www.iss.net/products_services/enterprise_protection/rsnetwork/sensor.php (current Aug. 2004).

[22] K. Julisch, "Mining Alarm Clusters to Improve Alarm Handling Efficiency," *Proceedings:* 17th Annual Computer Security Applications Conference (ACSAC'01), New Orleans, LA, Dec. 2001.

[23] K. Julisch and M. Dacier, "Mining Intrusion Detection Alarms for Actionable Knowledge", *Proceedings: 8th ACM International Conference on Knowledge Discovery and Data Mining*, Edmonton, Australia, Jul. 2002.

[24] K. Julisch, "Clustering Intrusion Detection Alarms to Support Root Cause Analysis," *ACM Transactions on Information and System Security,* vol. 6, no. 4, 2003, pp. 443 – 471.

[25] D. Kafura, "Generalization", http://people.cs.vt.edu/~kafura/cs2704/generalization.html, (current Dec. 2005).

[26] B. Kosko, "Fuzzy Cognitive Maps," *International Journal of Man-Machine Studies,* vol. 24, 1986, pp. 65-75.

[27] B. Kosko, *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence,* Prentice Hall, Englewood Cliffs, NJ, 1992.

[28] B. Kosko, *Fuzzy Engineering,* Prentice Hall, Upper Saddle River , NJ, 1997.

[29] M. A. Lee, "What is Soft Computing? *What is BISC?,"* http://www-bisc.cs.berkeley.edu/bisc/bisc.memo.html#what_is_sc (current Jan. 2004)

[30] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M.A. Zissman, "Evaluating Intrusion Detection Systems: The 1998 DARPA Off-Line Intrusion Detection Evaluation," *Proceedings: 2000 DARPA Information Survivability Conference and Exposition*, vol. 2, 2000.

[31] Z. Liu, S. M. Bridges, and R. B. Vaughn, "Combining Static Analysis and Dynamic Learning to Build Accurate Intrusion Detection Models," *Proceedings: IEEE International Information Assurance Workshop,* College Park, MD, Mar. 2005, pp. 164-177.

[32] M.L. Massie, B.N. Chun, and D.E. Culler, "The Ganglia Distributed Monitoring System: Design, Implementation, and Experience," *Parallel Computing*, 30 (7), Jul. 2004.

[33] S. Mathew, C. Shah, S. Upadhyaya, "An Alert Fusion Framework for Situation Awareness of Coordinated Multistage Attacks", *Proceedings: 3rd IEEE International Information Assurance Workshop (IWIA 2005)*, College Park, MD, Mar. 2005, pp. 95-104.

[34] M.I.T Lincoln Laboratory, "2000 DARPA Intrusion Detection Scenario Specific Data Sets," http://www.ll.mit.edu/IST/ideval/data/2000/2000_data_index.html (current Aug. 2004)

[35] M.I.T Lincoln Laboratory, "Lincoln Laboratory Scenario (DDoS) 1.0," http://www.ll.mit.edu/IST/ideval/data/2000/LLS_DDOS_1.0.html (current Aug. 2004)

[36] B. Morin and H. Debar. "Correlation of Intrusion Symptoms: an Application of Chronicles", *Proceedings: 6th International Conference on Recent Advances in Intrusion Detection (RAID'03),* Pittsburgh, PA, Sep. 2003.

[37] P. Ning, Y. Cui, and D. Reeves, "Constructing Attack Scenarios through Correlation of Intrusion Alerts," *Proceedings: ACM Conference on Computer & Communications Security*, Washington D.C., WA, Nov. 2002.

[38] P. Ning, D. Reeves, and Y. Cui, *Correlating Alerts using Prerequisites of Intrusions*, technical report TR-2001-13, Department of Computer Science, North Carolina State University , 2001.

[39] P. Ning, "TIAA: A Toolkit for Intrusion Alert Analysis," http://discovery.csc.ncsu.edu/software/correlator/ (current Aug. 2004)

[40] P. Ning, D. Xu, C. G. Healey, and R. A. S. Amant, "Building Attack Scenarios through Integration of Complementary Alert Correlation

Methods," *Proceedings: 11th Annual Network and Distributed System Security Symposium (NDSS '04)*, San Diego, CA, Feb. 2004.

[41] T. D. Ndousse, and T. Okuda. "Computational Intelligence for Distributed Fault Management in Networks using Fuzzy Cognitive Maps," *Proceedings: IEEE International Conference on Communications Converging Technologies for Tomorrow's Application,* New York, NY, 1996, pp. 1558-1562.

[42] Open Source Technology Group, Inc., "Tcpreplay: Pcap editing and replay tools for *NIX," http://tcpreplay.sourceforge.net/ (current Aug. 2005).

[43] C. F. Pelaez and J. B. Bowles. "Applying Fuzzy Cognitive Maps Knowledge Representation to Failure Modes Effects Analysis," *Proceedings: IEEE Annual Symposium on Reliability and Maintainability,* 1995, pp. 450-456.

[44] P. A. Porras, M. W. Fong, and A. Valdes, "A Mission-Impact-Based Approach to INFOSEC Alarm Correlation," *Proceedings: Recent Advances in Intrusion Detection*, Zurich, Switzerland. Oct. 2002.

[45] X. Qin and W. Lee, "Statistical Causality Analysis of INFOSEC Alert Data," *Proceedings: Recent Advances in Intrusion Detection,* Pittsburgh, PA, Sep. 2003.

[46] D. Rokos, "Multi-Sources Information Fusion for Satellite Images Classification", Technical Report, http://www.survey.ntua.gr/main/labs/rsens/DeCETI/IRIT/MSI-FUSION/ , current Dec. 2005).

[47] M. Roesch, "Snort-Lightweight Intrusion Detection for Networks," http://www.snort.org/docs/lisapaper.txt (current Jul. 2004).

[48] A Siraj, S. M. Bridges, and R. B. Vaughn, "Fuzzy Cognitive Maps for Decision Support in an Intelligent Intrusion Detection System," *Proceedings: International Fuzzy Systems Association/ North American Fuzzy Information Processing Society (IFSA/NAFIPS) Conference on Soft Computing,* Vancouver, Canada, Jul. 2001.

[49] A. Siraj, R. B. Vaughn and S. M. Bridges, "Intrusion Sensor Data Fusion in an Intelligent Intrusion Detection System Architecture." *Proceedings: Frameworks and Methods for the Study and Analysis of Trust in Information Systems: Minitrack in the Software Technology Track of the Thirty-Seventh Hawaii International Conference on System Sciences (HICSS-37)*, Hawaii, Jan. 2004.

[50] A. Siraj, R. B. Vaughn, and S. M. Bridges, "Decision Making for Network Health Assessment in an Intelligent Intrusion Detection System Architecture," *International Journal of Information Technology and Decision Making,* vol. 3, no. 2, 2004.

[51] E. Smith and J. H. P. Eloff. "Cognitive Fuzzy Modeling for Enhanced Risk Assessment in Health Care Institution," *IEEE Intelligent Systems and Their Applications*, March/April 2000, pp. 69-75.

[52] M. Steinbach, G. Karypis, and V. Kumar. "A Comparison of Document Clustering Techniques", KDD Workshop on Text Mining, 1999.

[53] D. Stylios and P. P. Groumpos, "A Soft Computing Approach for Modelling the Supervisor of Manufacturing Systems," *Journal of Intelligent and Robotics Systems*, vol. 26, no. 3-4, 1999, pp. 389-403.

[54] R. Taber, "Knowledge Processing with Fuzzy Cognitive Maps," *Expert Systems with Applications,* vol. 2, 1991, pp. 83-87.

[55] M. Torres, L. Zhen, G. Florez, R. B. Vaughn, and S. M. Bridges, "Attacking a High Performance Computer Cluster," *Proceedings: 15th Annual Canadian Information Technology Security Symposium*, Ottawa, Canada, May 2003.

[56] United States Department of Homeland Security, "Threats and Protection: Advisory System," http://www.dhs.gov/dhspublic/display?theme=29, (current Sep. 2004).

[57] A. Valdes and K. Skinner, "An Approach to Sensor Correlation," *Proceedings: 3rd International Symposium on Recent Advances in Intrusion Detection (RAID'03),* Toulouse, France, October 2000.

[58] A. Valdes and K. Skinner, "Probabilistic Alert Correlation," *Proceedings: 4th International Symposium on Recent Advances in Intrusion Detection (RAID'04)*, 2001.

[59] J. Q. Xin, J. E. Dickerson, and J. A. Dickerson, "Fuzzy Feature Extraction and Visualization for Intrusion Detection," *Proceedings: FUZZ-IEEE*, St. Louis, MO, 2003.

[60] N. Ye, J. Giordano, J. Feldman, and Q. Zhong, "Information Fusion Techniques for Network Intrusion Detection," *Proceedings: 1998 IEEE Information Technology Conference*, 1998.

[61] N. Ye and M. Xu, "Information Fusion for Intrusion Detection," *Proceedings: 3rd International Conference on Information Fusion*, Paris, France, Jul. 2000.

[62] J. Yen and R. Langari, *Fuzzy Logic: Intelligence, Control and Information,* Prentice Hall, Upper Saddle River, NJ, 1999.

[63] D. Yu and D. Frincke, "A Novel Framework for Alert Correlation and Understanding," *Proceedings: International Conference on Applied Cryptography and Network Security* (ACNS), Yellow Mountain, China, 2004.

[64] H.-J., Zimmermann, "Fuzzy Set Theory and its Applications", 3rd edition, Kluwer Academic Publishers, Norwell, MA, 1996.

[65] Y. Zhai, P. Ning, P. Iyer, and D. S. Reeves, "Reasoning about Complementary Intrusion Evidence", *Proceedings: 20th Annual Computer Security Applications Conference*, Dec. 2004, pp. 39-48.

# APPENDIX A

## COMPARATIVE SUMMARY OF THE LITERATURE REVIEW

Table A.1 Comparative Summary of the Literature Review

| WHO | WHEN | WHERE | OBJECTIVE | ALERT ASSOCIATION | PRIORITIZATION | CLUSTERING | CORRELATION | QUANTITATIVE ASSESSMENT | MECHANISM |
|---|---|---|---|---|---|---|---|---|---|
| Ye & Xu | 2000 | ASU | Fuse sensor results | N/A | X | X | X | Composite result | ANN, Linear & Logistic Regression" |
| Debar & Wespi | 2001 | IBM Zurich | Generate one alert per attack | Structural & Causal | ~ | √ | √ | Clusters | Expert Rules |
| Julisch | 2001 | IBM Zurich | Discover root causes | Structural | X | √ | X | X | Abstraction |
| Cuppens | 2001 | France Telecom | Identify global & synthetic alerts | Structural & Causal | X | √ | √ | X | Predicate Logic |
| Ning et al. | 2001 | NCSU | Discover Multi-staged attacks | Causal | X | √ | √ | X | Predicate Logic, Hyper Alert Graph |
| Mathew et al. | 2005 | SUNY | Discover Multi-staged attack scenarios | Causal | X | X | √ | Scenario | Scenario Graph |

Table A.1 Comparative Summary of the Literature Review (continued)

| WHO | WHEN | WHERE | OBJECTIVE | ALERT ASSOCIATION | PRIORITIZATION | CLUSTERING | CORRELATION | QUANTITATIVE ASSESSMENT | MECHANISM |
|---|---|---|---|---|---|---|---|---|---|
| Ye et al. | 1998 | ASU | Fuse sensor results | N/A | X | X | X | Composite result | Dempster-Shafer & Bayesian |
| Valdes & Skinner | 2000 | SRI International | Relate alerts that match closely but not perfectly | Structural | X | √ | ~ | X | Expert Rules & Bayes Formalism |
| Goldman et al. | 2001 | Honeywell Lab. | Discover overall intrusion situation awareness | Causal | √ | X | √ | X | Description Logic & Qualitative Probability Theory |
| Dain & Cunningham | 2002 | MIT Lincoln Lab | Discover scenarios between alerts | Structural & Causal | X | √ | √ | X | Expert Rules & Probabilistic Techniques |
| Yu & Frincke | 2004 | Mcrsft. Rsrh. | Model attacks | Causal | X | ~ | √ | Incidents | Hidden Colored Petri Net |

Table A.1 Comparative Summary of the Literature Review (continued)

| WHO | WHEN | WHERE | OBJECTIVE | ALERT ASSOCIATION | PRIORITIZATION | CLUSTERING | CORRELATION | QUANTITATIVE ASSESSMENT | MECHANISM |
|---|---|---|---|---|---|---|---|---|---|
| Siraj et al. | 2001 | MSU | Fuse alerts to provide an overall condensed view of system | Structural | X | √ | X | Host and user alet level | Fuzzy Cognitive Maps |
| Xin et al. | 2003 | ISU | Identify attacks from network features | Causal | X | X | √ | X | Fuzzy Cognitive Maps |
| Unified Alert Fusion Model | 2005 | MSU | Fuse alerts to provide an overall condensed view of system | Structural & Causal | √ | √ | √ | Host' overall security status | Fuzzy Cognitive Maps, Possibilistic Fusion Operators |

# APPENDIX B

## CATEGORIZATION OF ATTACKS FOR REALSECURE AND SNORT SENSORS

Table B.1 Categorization of Attacks for RealSecure and Snort Sensors

| Attack_Name | Alert_Type | Sensor |
|---|---|---|
| Admind | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| ATTACK-RESPONSES 403 Forbidden | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| ATTACK-RESPONSES directory listing | Active_Communication_Remote | Snort_Version_2.3.3 |
| ATTACK-RESPONSES Invalid URL | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| BAD-TRAFFIC loopback traffic | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| BAD-TRAFFIC tcp port 0 traffic | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| DDOS mstream agent to handler | Launch_Importation | Snort_Version_2.3.3 |
| DDOS mstream handler to agent | Launch_Importation | Snort_Version_2.3.3 |
| DNS_HInfo | Surveillance_Host | RealSecure_Version_7.0 |
| DNS_Query_All | Policy_Compliance_Informational | RealSecure_Version_7.0 |
| Echo_Reply_Without_Request | Policy_Compliance_Suspicious | RealSecure_Version_7.0 |
| Email_Almail_Overflow | Access_Violation | RealSecure_Version_6.0 |
| Email_Data | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Email_Debug | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| Email_Ehlo | Policy_Compliance_Informational | RealSecure_Version_6.0 |
| Email_From | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Email_ServerID | Probe_Service | RealSecure_Version_7.0 |
| Email_Subject | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Email_To | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Email_Turn | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| EventCollector_Info | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| EventCollector_Warning | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Finger_User | Probe_User | RealSecure_Version_7.0 |
| FTP_Filename | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| FTP_Get | Active_Communication_Remote | RealSecure_Version_7.0 |
| FTP_Pass | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| FTP_Port | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| FTP_Put | Active_Communication_Remote | RealSecure_Version_6.0 |
| FTP_Server_Identity | Probe_Service | RealSecure_Version_7.0 |
| FTP_Syst | Probe_Service | RealSecure_Version_6.0 |
| FTP_User | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| HTTP_ActiveX | Policy_Compliance_Suspicious | RealSecure_Version_6.0 |
| HTTP_Cisco_Catalyst_Exec | Access_Violation | RealSecure_Version_6.0 |
| HTTP_Get | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| HTTP_Get_Very_Long | Privilege_Violation | RealSecure_Version_7.0 |
| HTTP_GetArg | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| HTTP_Java | Policy_Compliance_Suspicious | RealSecure_Version_6.0 |
| HTTP_Server_ID | Probe_Service | RealSecure_Version_7.0 |
| HTTP_Shells | Active_Communication_Remote | RealSecure_Version_6.0 |
| HTTP_User_Agent | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| HTTP_Vulnerable_Client | Probe_Service | RealSecure_Version_7.0 |
| ICMP Destination Unreachable Port Unreachable | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| ICMP Echo Reply | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |

Table B.1 Categorization of Attacks for RealSecure and Snort Sensors (continued)

| Attack_Name | Alert_Type | Sensor |
|---|---|---|
| ICMP PING | Surveillance_Host | Snort_Version_2.3.3 |
| ICMP PING *NIX | Surveillance_Host | Snort_Version_2.3.3 |
| ICMP PING BSDtype | Surveillance_Host | Snort_Version_2.3.3 |
| ICMP redirect host | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| ICMP_Flood | Launch_Disruption | RealSecure_Version_7.0 |
| ICMP_Redirect | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Ident_User | Policy_Compliance_Informational | RealSecure_Version_7.0 |
| Image_GIF_CompressionError | Access_Violation | RealSecure_Version_7.0 |
| INFO FTP Bad login | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| INFO TELNET Bad Login | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| INFO TELNET Failed Login | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| INFO web bug 0x0 gif attempt | Policy_Compliance_Informational | Snort_Version_2.3.3 |
| LanMan_Share_Enum | Policy_Compliance_Informational | RealSecure_Version_7.0 |
| MS-SQL version overflow attempt | Privilege_Violation | Snort_Version_2.3.3 |
| Mstream_Zombie | Launch_Importation | RealSecure_Version_6.0 |
| Mstream_Zombie_Request | Launch_Importation | RealSecure_Version_7.0 |
| Mstream_Zombie_Response | Launch_Importation | RealSecure_Version_7.0 |
| NETBIOS NT NULL session | Policy_Compliance_Notification | Snort_Version_2.3.3 |
| Netbios_Session_Granted | Active_Communication_Remote | RealSecure_Version_7.0 |
| Netbios_Session_Request | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| NON-RFC HTTP DELIMITER | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| NTP_Time | Policy_Compliance_Informational | RealSecure_Version_7.0 |
| Ping_Sweep | Surveillance_Network | RealSecure_Version_7.0 |
| POLICY FTP anonymous login attempt | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| POP_Password | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| POP_Server_Identity | Probe_Service | RealSecure_Version_7.0 |
| POP_User | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Port_Scan | Probe_Service | RealSecure_Version_6.0 |
| RIPAdd | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| RIPExpire | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| RPC portmap sadmind request UDP | Policy_Compliance_Notification | Snort_Version_2.3.3 |
| RPC sadmind query with root credentials attempt UDP | Privilege_Violation | Snort_Version_2.3.3 |
| RPC sadmind UDP NETMGT_PROC_SERVICE overflow attempt | Privilege_Violation | Snort_Version_2.3.3 |
| RPC sadmind UDP PING | Probe_Service | Snort_Version_2.3.3 |
| RPC_CALLIT_Request | Access_Violation | RealSecure_Version_7.0 |
| RPC_Portmap_Getport | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| RSERVICES rsh root | Active_Communication_Remote | Snort_Version_2.3.3 |
| Rsh | Active_Communication_Remote | RealSecure_Version_6.0 |
| Sadmind_Amslverify_Overflow | Privilege_Violation | RealSecure_Version_6.0 |
| Sadmind_Ping | Probe_Service | RealSecure_Version_6.0 |
| SensorStatistics | Policy_Compliance_Notification | RealSecure_Version_7.0 |

Table B.1 Categorization of Attacks for RealSecure and Snort Sensors (continued)

| Attack_Name | Alert_Type | Sensor |
|---|---|---|
| SensorStatistics_Cumulative | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| SMB_Filename | Policy_Compliance_Informational | RealSecure_Version_7.0 |
| SNMP AgentX/tcp request | Privilege_Violation | Snort_Version_2.3.3 |
| SNMP public access udp | Policy_Compliance_Notification | Snort_Version_2.3.3 |
| SNMP request tcp | Probe_Service | Snort_Version_2.3.3 |
| SNMP request udp | Probe_Service | Snort_Version_2.3.3 |
| SNMP trap tcp | Probe_Service | Snort_Version_2.3.3 |
| SNMP_Activity | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| SNMP_Community | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| SNMP_Default_Backdoor | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| SSH_Detected | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| SSH_Version | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| Stream_DoS | Launch_Disruption | RealSecure_Version_6.0 |
| TCP_ACK_Ping | Policy_Compliance_Notification | RealSecure_Version_7.0 |
| TCP_Probe_DNS | Probe_Service | RealSecure_Version_7.0 |
| TCP_Probe_HTTP | Probe_Service | RealSecure_Version_7.0 |
| TCP_Probe_Proxy | Probe_Service | RealSecure_Version_7.0 |
| TCP_Probe_SMTP | Probe_Service | RealSecure_Version_7.0 |
| TCP_Urgent_Data | Policy_Compliance_Suspicious | RealSecure_Version_6.0 |
| TELNET access | Active_Communication_Remote | Snort_Version_2.3.3 |
| TELNET login failure | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| TELNET login incorrect | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| Telnet_authintication_failed | Policy_Compliance_Suspicious | RealSecure_Version_7.0 |
| Telnet_Login | Active_Communication_Remote | RealSecure_Version_7.0 |
| TelnetEnvAll | Active_Communication_Remote | RealSecure_Version_6.0 |
| TelnetTerminaltype | Policy_Compliance_Notification | RealSecure_Version_6.0 |
| TelnetXdisplay | Active_Communication_Remote | RealSecure_Version_6.0 |
| UDP_Port_Scan | Probe_Service | RealSecure_Version_6.0 |
| UDP_Probe_DNS | Probe_Service | RealSecure_Version_7.0 |
| UDP_Probe_Other | Probe_Service | RealSecure_Version_7.0 |
| WEB-ATTACKS id command attempt | Probe_User | Snort_Version_2.3.3 |
| WEB-CGI campus access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-CGI count.cgi access | Access_Violation | Snort_Version_2.3.3 |
| WEB-CGI db2www access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-CGI finger access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-CGI icat access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-CGI redirect access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-CGI wrap access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-FRONTPAGE /_vti_bin/ access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-IIS %2E-asp access | Policy_Compliance_Suspicious | Snort_Version_2.3.3 |
| WEB-IIS fpcount access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-IIS fpcount attempt | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-IIS iissamples access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-IIS register.asp access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-MISC /doc/ access | Privilege_Violation | Snort_Version_2.3.3 |

Table B.1 Categorization of Attacks for RealSecure and Snort Sensors (continued)

| Attack_Name | Alert_Type | Sensor |
|---|---|---|
| WEB-MISC backup access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-MISC counter.exe access | Privilege_Violation | Snort_Version_2.3.3 |
| WEB-MISC RBS ISP /newuser access | Privilege_Violation | Snort_Version_2.3.3 |
| Windows_Null_Session | Policy_Compliance_Notification | RealSecure_Version_7.0 |

# APPENDIX C

# CRITICALITY INDEXES FOR SOURCE/TARGET COMMUNICATION AND ATTACKS

Table C.1 Criticality Indexes for Source/Target Communication

| Source/Target | Criticality |
|---|---|
| Outside_Source_Inside_Target | 1.00 |
| Inside_Source_Outside_Target | 1.00 |
| Inside_Source_Inside_Target | 0.75 |
| Outside_Source_Outside_Target | 0.25 |

Table C.2 Criticality Indexes for Attack

| Attack | Criticality |
|---|---|
| Acess_Violation | 0.50 |
| Active_Communication_Local | 0.75 |
| Active_Communication_Remote | 0.75 |
| Launch_Disruption | 1.00 |
| Launch_Importation | 1.00 |
| Launch_Modification | 1.00 |
| Launch_Revelation | 1.00 |
| Policy_Compliance_Informational | 0.05 |
| Policy_Compliance_Notification | 0.05 |
| Policy_Compliance_Suspicious | 0.05 |
| Privilege_Violation | 0.50 |
| Probe_Service | 0.25 |
| Probe_User | 0.25 |
| Surveillance_Host | 0.10 |
| Surveillance_Network | 0.10 |