發掘可疑網路行爲的聯合防禦分析方法

研究生：林永彧　　　　　　　　　　　指導教授：曾憲雄　博士

國立交通大學資訊學院

資訊科學與工程研究所

摘要

隨著網路入侵工具的快速普及，網路入侵事件的型態也逐漸改變。參考最新的網路安全威脅報告，網路入侵行為正朝向隱密化與目標特定化而演進。許多研究已經針對底層的網路資料分析網路入侵行為，例如入侵偵測系統（IDS）；然而這些方法可能產生數量龐大的錯誤警報，要從這些含錯誤警報的資料中找到有用的資訊，管理者須具備相關的經驗或知識。為了減輕管理者的負擔，必須先系統化擷取出有用的未知攻擊序列，再由管理者進行主機修復與攻擊事件研究。然而每種不同的攻擊都有自己的特性，目前並沒有任何單一方法可以完美的分析網路警報而同時找出實際的多種入侵。在這篇論文中，我們提出一個基於聯合防禦概念的可疑網路行為探勘知識（CDSNB）架構。這個架構主要包含三個階段的演算過程：分別是資料前處理階段，警報過濾階段及聯合分析階段。資料前處理階段被用來區分符合某些特定條件的主機，作為聯合分析階段的目標主機群組；此外，警報資料依據聯合分析階段的需求，被轉換成特定的資料格式。因為充斥錯誤警報，警報過濾階段便藉由建立警報的過濾模型（FM），藉此過濾多數的錯誤警報，以作為聯合分析階段的可靠資料來源。聯合分析階段則是從多台具有特定條件主機的觀點，分析各種攻擊模式，並將

結果轉化為容易分析的格式提供管理者作為參考。在這個知識導向的分析架構下，系統與管理者不斷進行互動，彈性的協助管理者進行各階段適當的演算法決策。最後，管理者可藉由經過整合的可疑入侵資訊，進行事件防禦或是修復主機弱點，甚至追溯攻擊起源。因此，我們希望可以藉此達到有效預防攻擊，並準確發掘新的攻擊模式，並同時減低管理者在分析階段的負擔。

關鍵字：聯合防禦、入侵偵測、知識導向、警報

A Study of Collaborative Discovering of Suspicious Network Behaviors

Student: Yung-Yu Lin   Advisor: Dr. Shian-Shyong Tseng

Institute of Computer Science and Engineering
National Chiao Tung University

Abstract

As the rapid growth of network attacking tools, patterns of network intrusion events change gradually. Referring to the newest Symantec Internet Security Threat Report, we found that network intrusion behaviors evolve into more hidden and target-specific behaviors. There are many researches had been proposed to analyze network intrusion behaviors in accordance with low-level network data. However, since these researches might suffer a large mount of false alerts, it is very difficult for network administrators to discover useful information from these alerts. To reduce the load of administrators, by collecting and analyzing unknown attack sequences systematically, administrators can do the duty of fixing the root causes and researching attack events. However, due to the different characteristics for each intrusion, there is no single analysis method which can correlate IDS alerts perfectly and discover all kinds of real intrusion patterns up to the present. Therefore, a knowledge-based framework for **Collaborative Discovering Suspicious Network Behaviors (CDSNB)** is proposed in this thesis. The framework of CDSNB consists of three phases: **Data Preprocessing Phase**, **Alert Filtering Phase** and **Collaborative Analysis Phase**. The Data Processing Phase is used to divide sensors into groups with specific system and network profiles, and IDS alerts of these groups are transformed into alert transactions with specific data formats according to requirements in the Collaborative Analysis Phase. Because of numerous of false alerts, the Alert Filtering Phase is used to construct **Filter Model (FM)** of sensors in specific group to filter most false alerts. The Collaborative Analysis Phase is used to analyze each alert pattern and classify the results into aggregated information for administrators as references of intrusion defense in the viewpoint of specific sensor groups with similar backgrounds and behaviors. In this knowledge-based analysis framework, the system interacts with administrators to assist them making appropriate decisions in each phase. According to the urgent situations of different levels, Network administrators can do event protecting or vulnerability repairing, even or cause tracing of attacks. Therefore, the knowledge-based framework of CDSNB can prevent attacks effectively, find novel attack patterns exactly and reduce the load of administrators efficiently.

Keywords: Collaborative Defense, Intrusion Detection, Knowledge-Based, IDS Alerts

誌謝

這篇論文的完成，必須感謝許多人的協助與支持。首先必須感謝我的指導教授，曾憲雄博士，由於他耐心的指導與勉勵，讓我得以順利完成這兩年內的研究與學業，受益匪淺，感激不盡。同時也感謝我的口試委員，陳年興教授、黃國禎教授與胡毓志教授，給予我相當多寶貴的建議，讓這篇論文更有價值。

感謝三位博士班的學長，王慶堯學長、林順傑學長和曲衍旭學長，兩年的指導期間，讓我學習到許多理論知識與實務技巧，並給予我這篇論文許多寶貴的意見，解答了我的疑惑，並不厭其煩的協助我論文的修改工作，深表感激。

同時也感謝實驗室的其他同窗夥伴，施南極、蔡昇翰、林喚宇、羅仁杰、葉展彰、張晉瑲、任珍妮以及鄧嘉文，不論是學業上或是生活上，互相扶持的度過這段既忙碌又充實的碩士生涯。另外感謝實驗室的其他學長、學弟及在職專班同學們，因為攜手合作，讓我在知識工程實驗室的歲月更添完整與豐富。

深深感謝我的母親，在背後默默支持我度過我的碩士生涯，讓我在面對挫折的時候能繼續向前。最後，我要感謝女朋友盈潔對我的支持與鼓勵，成為我的支柱，讓我能在這段期間無後顧之憂的完成碩士學業。

要感謝的人很多，無法一一詳述。感謝您們，每一位曾經幫助我的人。

Table of Content

List of Figures

List of Algorithms

Chapter 1: Introduction

With the rapid development of Internet, the Internet is becoming more and more complicated which results in the security on Internet becoming one of the most important issues today. Since many insecure network segments in Internet can still be compromised for different intensions, many intrusions such as probing, user to root (U2R), remote to local (R2L), Denial of Service (DoS) and Rootkits which may threaten Internet service providers seriously have been proposed. All of these intrusions could be treated as anomaly network behaviors.

The survey [21] of year 2006 shows a significant trend of evolution and difference between present and traditional intrusions. In the past, attackers were desirous of showing their extraordinary computer skills to meditate malicious programs to induce large-scale depredations of Internet to go down in history in the computer domain. The situation is very different nowadays, and behaviors of intrusions have become more and more variable and rapid. Figure 1.1 can indicate which industries are more frequent targets of focused attacks. This metric may be affected by the overall attack rate experienced in each industry; nevertheless, it provides an indication of the interest that an industry holds for targeted attackers. Since intrusions are more and more target-specific, attackers may reap profits by stealing secret information from specific victims. As shown in Figure 1.1, it is common for attackers to steal account information of clients in these industries of Accounting and Small business since such secret data are worth to other criminal gangs. Rootkits is one kind of these present concealed intrusions.

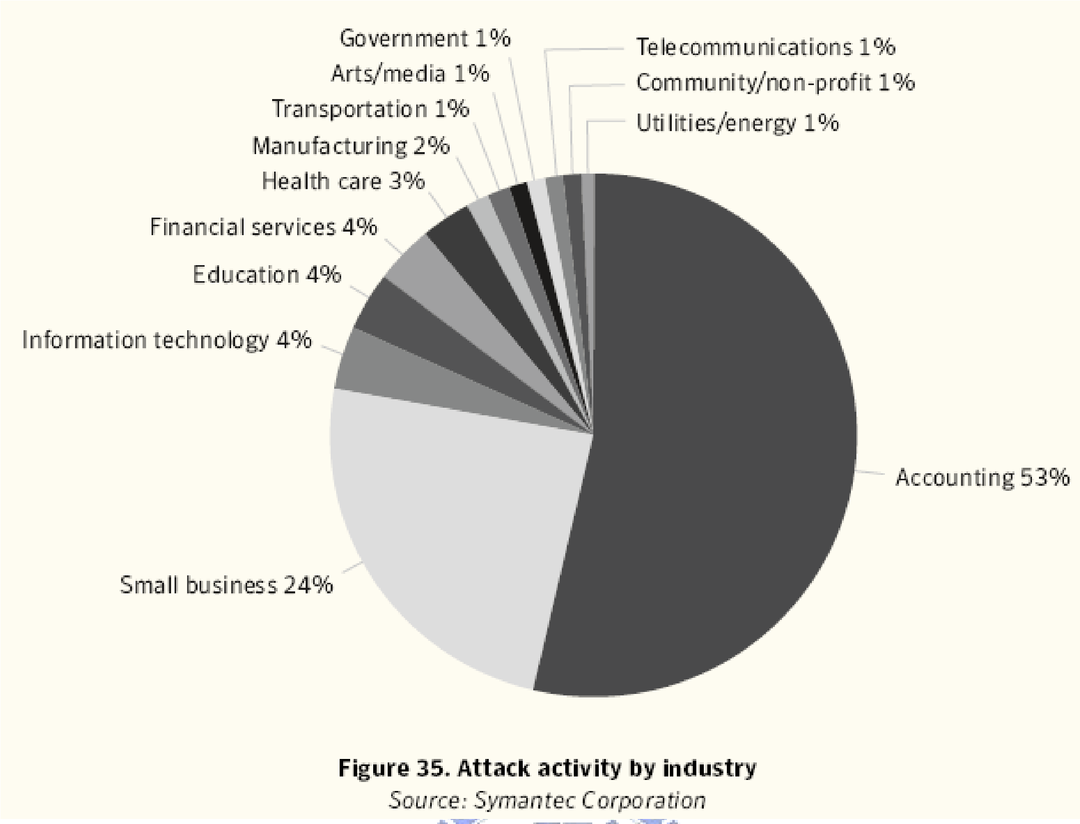


Figure 1.1: Attack activity by industry.

In order to detect and prevent anomaly network behaviors, many Intrusion Detection Systems (IDS) or Firewalls have been developed to focus on well-known intrusion patterns through packet-based information, connection-base information, or some statistical network information. Although these kinds of approaches can be useful to defend the obvious activity patterns of intrusions, many intrusions are still hard to be detected by IDSs to notice human experts because of numerous noises and complex information among different intrusions. It is very difficult for experts or administrators to generalize useful and indeed intrusion patterns from the dirty information.

Although some researcher proposed many artificial intelligence methods such as generic algorithm, neural networks, and data mining approaches, to discover either unknown or useful patterns for experts, lots of hidden and concealed intrusion patterns may still be escaped from these approaches because of insufficient and dirty information. There is no one analysis method which can discover intrusion patterns perfectly from IDS information data. Therefore, we are concerned with how to design a systematic framework to assist administrators discovering intrusion patterns with IDS information data, or called IDS alerts.

Among the systematic approaches proposed by previous researchers to analyze IDS alerts, the analysis processes are usually pre-defined and different analysis methods are used in different researches. Some researchers have discussed how to get better analysis results in some special cases using appropriate analytical methods. Out idea is to integrate multiple analysis methods to get different analytical results, and previous analytical researches can be applied and integrated. Since similar concepts, motivations, and sub-procedures in most researches, a **Knowledge-based framework for Collaborative Discovering of Suspicious Network Behaviors** is proposed in this thesis to integrate most analytical algorithms which is used for alert transformation, alert correlation, alert aggregation and alert filtering. With the knowledge-based approach, it is possible to assist administrators selecting appropriate methods for different requirements and to easily to replace original analytical algorithms with new methods provided by other experts. Besides, integrated alert transactions can be analyzed on desired concept levels of multiple dimensions in the data cube for discovering useful intrusion patterns with OLAP and data warehouse technique.

The proposed framework consists of three phases: **Data Preprocessing Phase, Alert Filtering Phase and Collaborative Analysis Phase**. In Data Processing Phase, a **Sensor Grouping (SG)** algorithm is proposed to assist administrators dividing sensors into groups with specific system and network profiles, and these groups will be used in the Collaborative Analysis Phase; besides, an **Alert Format Transformation (AFT)** algorithm is proposed to assist administrators transforming IDS alerts of these groups into alert transactions with specific appropriate data formats according to requirements in the Collaborative Analysis

Phase. Because of numerous of false alerts, an **Alert Filtering Method Selection (AFMS)** algorithm in Alert Filtering Phase is proposed to assist administrators constructing a appropriate **Filter Model (FM)** of sensors in specific group to filter most false alerts, and the results of this phase are reliable data sources for the Collaborative Analysis Phase. In Collaborative Analysis Phase, an **Intra-Group Collaborative Analysis Selection (IGC-A)** algorithm is proposed to assist administrators analyzing each alert patterns with appropriate methods for specific attack types in one specific sensor group; finally, an **Inter-Group Collaborative Behavior Sharing (IGC-B)** algorithm is proposed to classify the results into aggregated information for administrators to select the appropriate intrusion defense in the viewpoint of specific sensor groups with similar backgrounds and behaviors.

To verify the feasibility of this knowledge-base framework, the corresponding algorithms in each phase and the data set as our data source are proposed and used to test the performance of these algorithms. As shown in Chapter 7, we can obtain useful information of suspicious alert patterns about novel intrusions.

The main contributions of this thesis are listed below:

(1) Using a knowledge-based approach to assist administrators exporting IDS alerts to discover suspicious patterns of novel attacks for intrusion detection.

(2) Merging multiple analytic methods by a common framework for increasing the diversity of intrusion detection analysis.

(3) Construct concept hierarchies of IDS alert transactions for discovering network intrusions cross every concept level of each dimension.

Chapter 2: Related Work

2.1 Traditional Analysis Approaches for Network Intrusion

As the cost of the information processing and Internet accessibility falls, more and more organizations are becoming vulnerable to a wide variety of cyber threats. According to a recent survey by CERT/CC (Computer Emergency Response Team/Coordination Center), the rate of cyber attacks has been more than doubling every year in recent times. It has become increasingly important to establish our information systems, especially those used for critical functions in the military and commercial sectors, resistant to and tolerant of such attacks.

Intrusion detection includes identifying a set of malicious actions that compromise the integrity, confidentiality, and availability of information resources. Traditional methods for intrusion detection are based on extensive knowledge of signatures of known attacks, where monitored events are matched against the signatures to detect intrusions. These methods extract features from various audit streams, and detect intrusions by comparing the feature values to a set of attack signatures provided by human experts. The signature database has to be manually revised for each new type of intrusion that is discovered. A signature limitation of signature-based methods is that it is hard to detect emerging cyber threats, since by their very nature these threats may be launched using previously unknown attacks. These limitations have led to an increasing interest in intrusion detection techniques based upon data mining.

Previous researchers have developed systematic approaches to analyze network traffic [3], [9], [18], [16] and the format of network traffic is usually pre-defined and hard to change. Continuous Query systems [4], [12] share many of the concerns of acquiring and filtering continuous streams of data from the database field, but do not have the ability to easily add new function over that data.

2.2 Using OLAP for Log Analysis

OLAP (On-Line Analytical Processing) can organize and present data in various formats in order to accommodate the diverse needs of the different analysis approaches. OLAP server provides server operations for analyzing multidimensional data cube:

(1) Roll-up: The roll-up operation collapses the dimension hierarchy along a particular dimension(s) so as to preset the remaining dimensions as a coarser level of granularity.

(2) Drill-down: In contrast, the drill-down function allows users to obtain a more detailed view of a given dimension.

(3) Slice: Here, the objective is to extract a slice of the original cube corresponding to a single value of a given dimension. No aggregation is required with this option. Instead, server allows the user to focus on desired values.

(4) Dice: A related operation is the dice. In this case, users can define a sub-cube of the original space. In other words, by specifying value ranges on one or more dimensions, the user can highlight meaningful blocks of aggregated data.

(5) Pivot: The pivot is a simple but effective operation that allows OLAP users to visualize cube values in more natural and intuitive ways.

A specific implementation of using OLAP technology on log analysis was discussed in [6]. The OLAP architecture is flexible in analyzing data; however only single data source is used in this architecture. Data source is limited to Windows NT system log and concept hierarchies are pre-defined. The diversity of data source and the quality of concept hierarchies would affect the ability of analysis. A Network Intrusion Monitoring System Architecture based on OLAP is proposed in [21] to integrate multiple network traffic data sources. Various systematic analysis approaches can be applied through OLAP server using operations such as drill-down, roll-up, slicing, etc., and OLAP Mining (OLAM) is then used to increase the diversity of network analysis result. Through Network Intrusion Monitoring System (NIMS), multiple data sources can be integrated to increase diversity of analysis approaches. Integrated data source can be analyzed on different dimensions and different concept levels to get more information.

2.3 IDS Alert Aggregation

Intrusion Detection Systems (IDSs) are widely deployed in computer networks to stand against a wide variety of attacks. IDSs are considered as powerful security tools in computer systems environments. These systems collect activities within the protected network and analyze them in order to detect intrusions. System activities are usually collected from two main sources, network packet streams and host log files. Once the information is collected, the detection algorithm starts looking for any evidence for intrusion existence. That makes it possible for security experts to defend intrusions quickly originally; but the number of alerts increase rapidly nowadays, and it makes it more difficult to organize information of intrusions from these numerous alerts. There are many researches discussing about information aggregation and correlation of different alerts.

In [22] a probabilistic-based reasoning method is used to correlate alerts by measuring and evaluating the similarities of alert attributes. Alert aggregation and scenario construction are conducted by enhancing or relaxing the similarity requirements in some attributes fields. In [10] a correlation system based on Bayesian reasoning is proposed. The system predefined the relationship between mission goals and corresponding security events for further inference and correlation.

In [17] a “mission-impact-based” correlation system with a focus on the attack impacts on the protected domains. The system uses clustering algorithms to aggregate and correlate alerts. Security incidents are ranked based on the security interests and the relevance of attacks to the protected networks and systems. Backward and forward reasoning techniques which are applied to correlate alerts are applied with duplicate and consequence relationship in [8]. They use clustering algorithms to detect attack scenarios and situations. This approach pre-defines consequences of attacks in a configuration file.

In [13] chronicle formalism is applied to aggregate and correlate alerts. The approach performs attack scenario pattern recognition based on known malicious event sequences. Therefore, this approach is similar to misuse detection and cannot detect new attack sequences.

There are many researches which build alert correlation systems based on matching the pre-/post-conditions of individual alerts [14], [7] and [5]. The idea of this approach is that prior attack steps prepare for later ones. Therefore, the consequences of earlier attacks correspond to the prerequisites of later attacks. The correlation engine searches alert pairs that have a consequence and prerequisite matching. Further correlation graphs can be built with such alert pairs [14]. One challenge to this approach is that a new attack cannot be paired with any other attacks because its prerequisites and consequences are not defined. Recently, some authors of this approach have extended the pre/post-condition-based correlation technique to correlate some isolated attack scenarios by hypothesizing missed attack steps [15].

2.4 IDS Alert Reduction

Recently, IDSs deployment raises a serious problem, namely managing of a large number of triggered alerts. This problem becomes worse by the fact that some commercial IDSs may generate thousands of alerts per day. Most important of all, many researchers indicate that most of alerts are false positive alerts, and this situation arises from the characteristics of IDSs. Security experts design IDSs for detecting intrusions more powerfully, but that makes IDSs more sensitive simultaneously. Identifying the real alerts from the huge volume of alerts is a frustrating task for security experts or network administrators. Thus, reducing false alerts becomes a critical issue in IDSs efficiency and usability.

Only few researches have been done on reducing false alerts of IDSs. A filtration technique of mining historical alerts to reduce false alerts rate is proposed [1]. The basic idea of this research is: Frequent behavior, over an extended period of time, is likely to be normal. First, an approach is proposed for characterizing the "normal" stream of alerts. In addition, an algorithm for detecting anomalies by using continuous and discontinuous sequential patterns is developed, and the results of preliminary experiments shows this research is indeed effective for some cases of real-world intrusions.

In [19], it proposes a false alert classification model to reduce the false alert rate using classification analysis of data mining techniques. The model was implemented based on associative classification in the domain of DDoS attack. This research presents applying decision tree to reduce false alerts from IDS and improve the performance of IDSs for keeping important information. Else, a probabilistic approach is introduced for the coupled sensors to reduce the false alerts in [22].

Chapter 3: Knowledge-Based Framework for Collaborative Discovering of Suspicious Network Behaviors

3.1 Issues for Discovering Suspicious Network Behaviors

Traditional analysis methods use different data sources according to different methods of IDS alert analysis. These methods analyze alerts by their own pre-defined data formats. Different researches use different analysis methods to get the optimal analysis results. There are some different characteristics among some analysis methods, and it is possible to cause different efforts because of different concept hierarchies.

There is a conceptual deficiency in these researches, and it is that each method is totally independent analysis between others; some of them are provided with complete data formats, and some are conspicuous on analysis performance. To integrate advantages of different methods, redesigning a new analysis algorithm is the only one way. Besides, it is possible for administrators to have their own appropriate analysis methods according to their domain knowledge and experiences. Difficulties of transformation and shortage of choices between different methods are the main deficiencies of integrating large amount of alerts effectively and efficiently. If administrators can select their own appropriate analysis procedures with diverse data sources, data formats and analysis algorithms, they will get more valuable results in analyzing IDS alerts. Moreover, if administrators can transform their new ideas of intrusion detection into effective detection algorithms easily, it will make collaborative defense systems will become more meaningful and powerful.

The main purposes of IDS alerts collection and analysis are as followings. First, finding more meaningful alert information is used to avoid being affected by large amount of false alerts. Second, discovering the information relation between these real alerts is used to verify system vulnerabilities and to infer attack causes. Some issues are derived from these purposes:

(1) How to choose appropriate analysis targets and data formats.

(2) How to filter false alerts efficiently.

(3) How to discover attack patterns and display appropriate data types for administrators to make policies.

Our concept is using a knowledge-based architecture to aggregate processes of analysis methods and to generalize most part of analysis algorithm architectures. Administrators can select appropriate methods for specific attack types to analyze alerts and to discover suspicious attack patterns.

3.2 The Knowledge-Based Framework

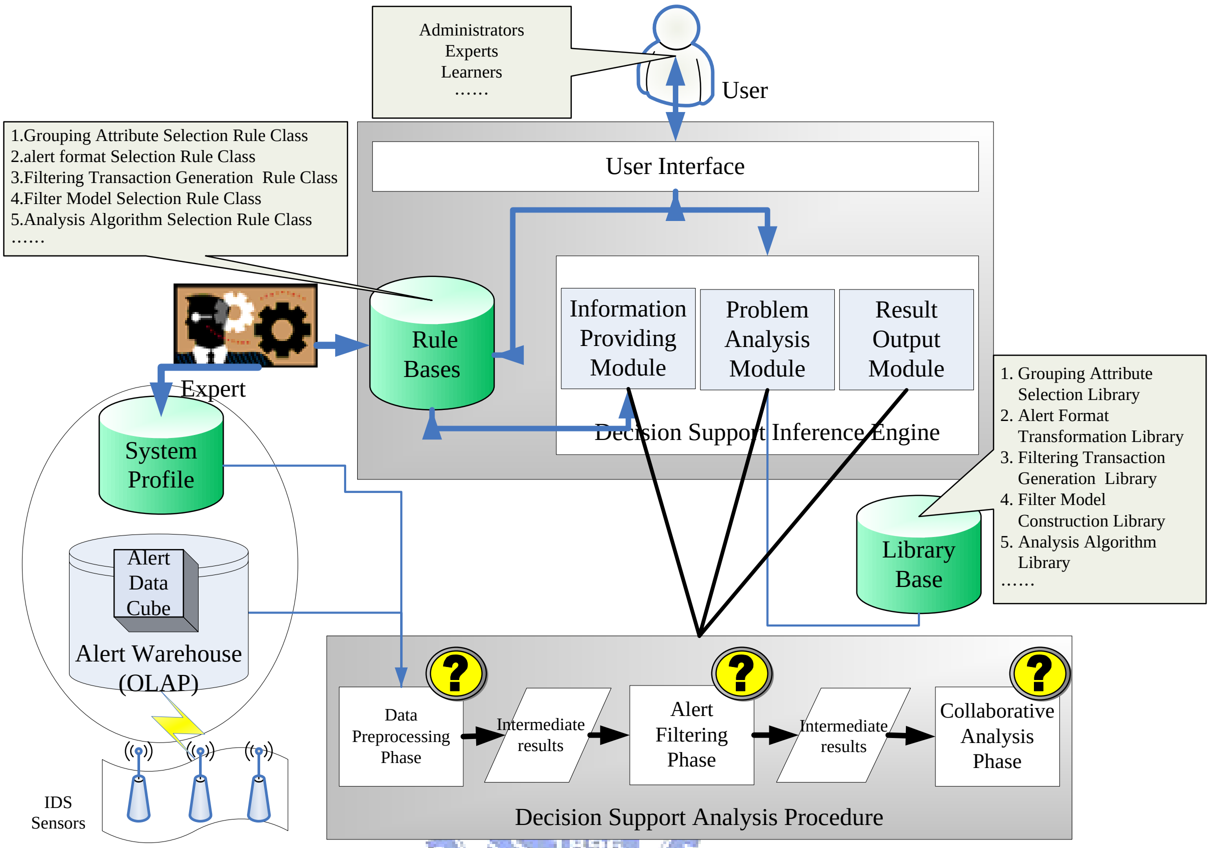

Administrators
Experts
Learners
……
User
1.Grouping Attribute Selection Rule Class
2.alert format Selection Rule Class
3.Filtering Transaction Generation Rule Class
4.Filter Model Selection Rule Class
5.Analysis Algorithm Selection Rule Class
……
User Interface
Rule Bases
Information Providing Module
Problem Analysis Module
Result Output Module
Decision Support Inference Engine
Expert
System Profile
Alert Data Cube
Alert Warehouse (OLAP)
1. Grouping Attribute Selection Library
2. Alert Format Transformation Library
3. Filtering Transaction Generation Library
4. Filter Model Construction Library
5. Analysis Algorithm Library
……
Library Base
IDS Sensors
Data Preprocessing Phase
Intermediate results
Alert Filtering Phase
Intermediate results
Collaborative Analysis Phase
Decision Support Analysis Procedure

Figure 3.1: Knowledge-based Framework for Collaborative Discovering of Suspicious Network Behaviors.

Before designing the analysis procedure, a system framework of knowledge-based approach shown in Figure 3.1 is proposed. It consists of three components: first, IDS sensors are responsible for triggering network unusual alerts of each host, and then forwarding these alerts to alert warehouse of local server to store alert information transactions. Besides, administrators construct system profile databases of all hosts in their subnets manually, and modify the content of the database if there is any host being changed its system. The system maintains rules of each analysis phases provided by security experts initially, and store them in knowledge base. These rules are used to interact with administrators for to making policies of analysis methods according to target attack types, and administrators can modify the content of knowledge base according to their requirements. With ample data sources and policy rules, it is possible for the system to assist administrators to implement analysis procedure.

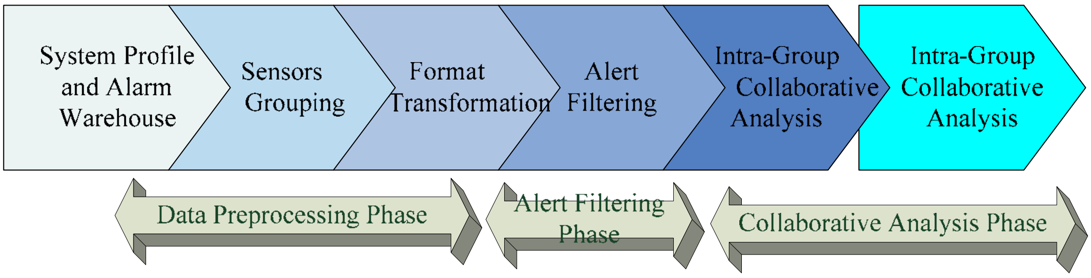

System Profile and Alarm Warehouse
Sensors Grouping
Format Transformation
Alert Filtering
Intra-Group Collaborative Analysis
Intra-Group Collaborative Analysis
Data Preprocessing Phase
Alert Filtering Phase
Collaborative Analysis Phase

Figure 3.2: Flow Chart of Analysis Procedure.

In order to solve the problems stated in Section 3.1, an analysis procedure shown in Figure 3.2 with three phases is proposed. It consists of: Data Preprocessing Phase, Alert Filtering Phase and Collaborative Analysis Phase.

In Data Preprocessing Phase, this system must provide ample system and alert information to conform to requirements of all analysis methods, so we redesign schemas of system profile database and alert warehouse. Moreover, a Sensor Grouping Query (SGQ) algorithm is proposed to assist administrators to choose appropriate attributes in data sources and information formats of collaborative defense sensors, it is used to divide sensors into different groups according to their specific profiles. At last, an Alert Format Transformation (AFT) algorithm is proposed to transform raw data in alert warehouse into specific data formats according to requirements of administrators.

In Alert Filtering Phase, it is necessary to filter false alerts in order to reduce the affect of noise on analysis results as more as possible. An Alert Filtering Model Selection (AFMS) algorithm is proposed to interact with administrators to assist administrators choosing appropriate filter methods, so that dirty data become clear data with more meaning.

In Collaborative Analysis Phase, administrators are asked to choose appropriate analysis methods from the analysis algorithm library according to the requirement of administrators. An Intra-Group Collaborative Analysis Selection (IGC-A) algorithm is proposed to interact with administrators to get better analysis results to provide references of problem solving information. Finally, an idea of integration between several analysis results and design of information sharing module is proposed to aggregate and exchange specific attack patterns. In this thesis, it is named as Inter-Group Behavior Sharing (IGBS) algorithm.

3.3 The Scheme of System Profile and Alerts

The constructions of database and data warehouse are used to record the information of system profiles and IDS alerts, and these will be useful for transformation between multi-dimensions analysis methods. The system profile database is constructed manually by experts at the very start, and modified by administrators if there is any new system added or any system changed. The IDS alert warehouse is used to collect IDS alert information from sensors, and transformed into large amount of alert transactions, such as Snort sensor alerts. The issue of alert warehouse from different types of IDS sensors has already been discussed by many researchers, so it is not our concern in this thesis. According to our assumption, IDS alert warehouse collects alert transactions form sensors with the same type of IDS sensors. In our thesis, we revise the design of system profile and alert warehouse in [11] to match the requirement of our research. System profile is just used to record information of hosts and networks without frequent changes, so the function of database is sufficient for this requirement. Besides, data warehouse is used to record alert information and to analyze complex alert information among different dimensions.

(1) The Schema of System and Network Profile Database

A new system profile database schema is proposed in Figure 3.3 to record host information according to the requirement of this thesis. The explanations of the tables are as followings: 'Local Server' table is used to store system information of the only local server; 'Open Service' table is used to record what kind of service the server provides; 'Local Profile' table is used to store information of network profiles and sensor scales in this subnet; 'Sensor' table is used to record detailed information of each sensor; 'Administrator' table is used to store information of administrators; 'Network' table is used to record scales under this subnet.

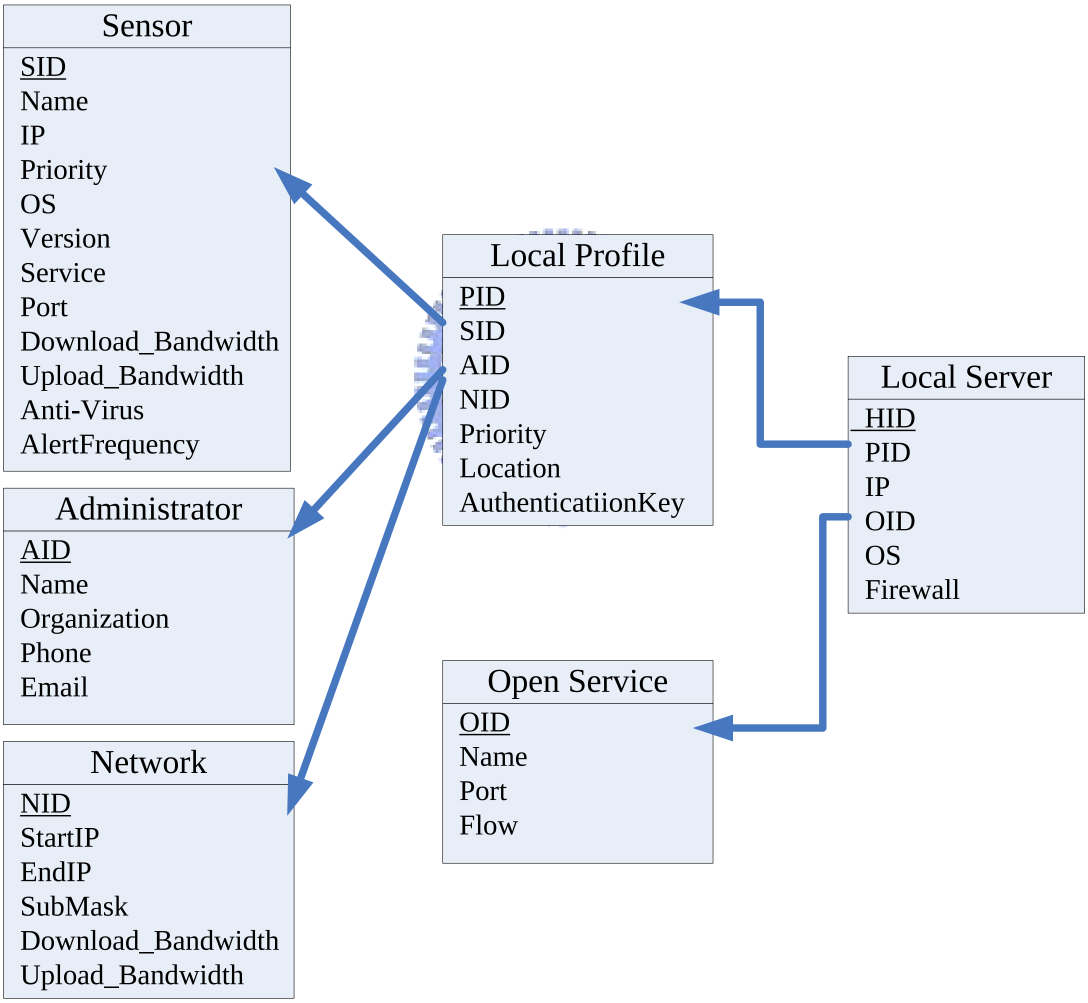
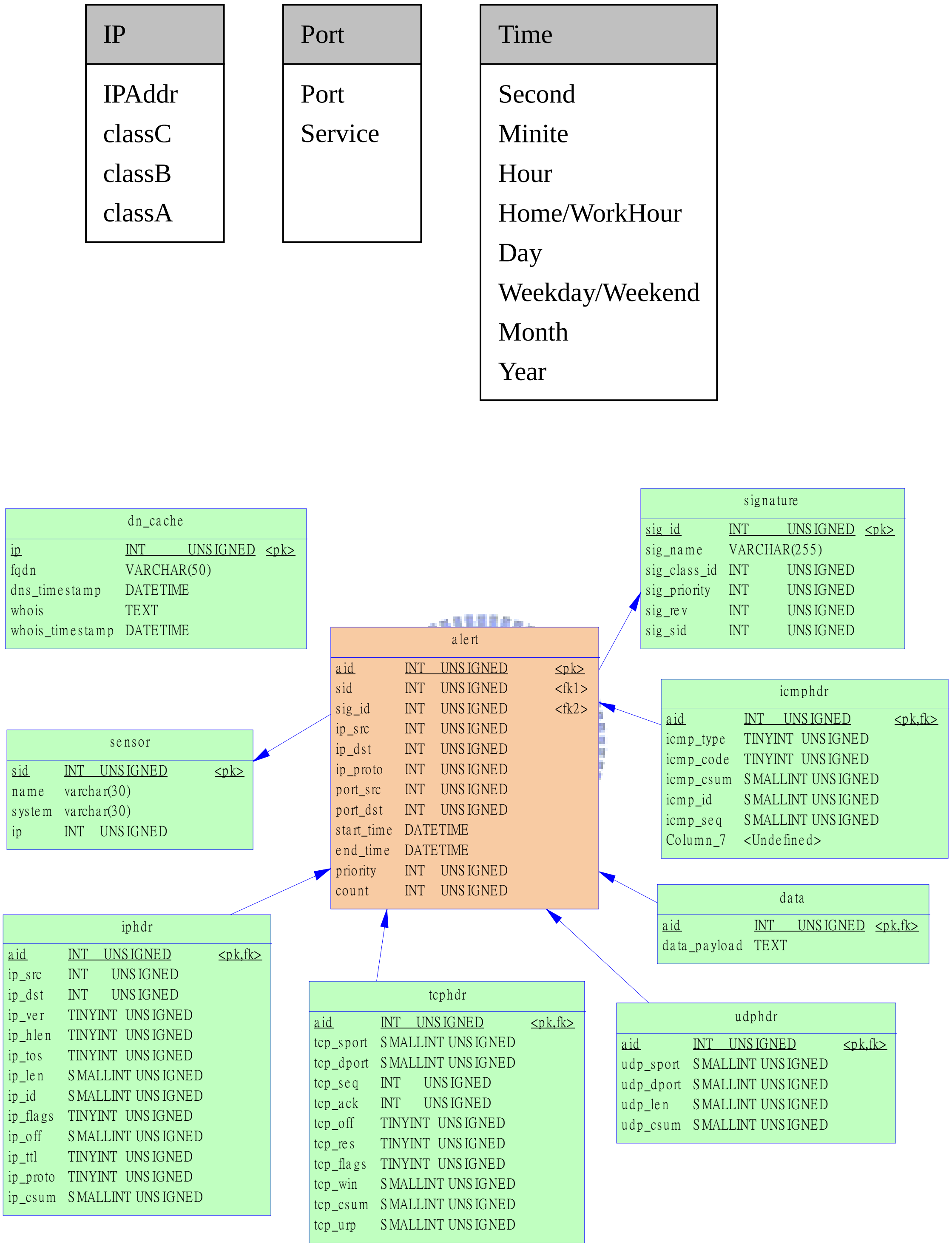


Figure 3.3: The schema of system and network profiles.

(2) The Schema of Alert Warehouse

At first, the special design of OLAP data warehouse proposed in this thesis must be highlighted to differentiate from traditional data warehouse schema. In all facts related to IP, Port or time, we have an idea that the characteristic of concept hierarchy should be emphasized to provide administrators better choices with more dimensions, so that three kinds of dimension tables are redesigned as follows. These dimension tables are corresponding to all IP-related, Port-related and Time-related facts in data warehouse schema of alerts. Moreover, there are many researches which discuss the issue of alert warehouse schema, so we continue using the schema of one research [11] shown in Figure 3.4.



IP
IPAddr
classC
classB
classA

Port
Port
Service

Time
Second
Minite
Hour
Home/WorkHour
Day
Weekday/Weekend
Month
Year

dn_cache		
ip	INT UNSIGNED	<pk>
fqdn	VARCHAR(50)	
dns_timestamp	DATETIME	
whois	TEXT	
whois_timestamp	DATETIME	

signature		
sig_id	INT UNSIGNED	<pk>
sig_name	VARCHAR(255)	
sig_class_id	INT UNSIGNED	
sig_priority	INT UNSIGNED	
sig_rev	INT UNSIGNED	
sig_sid	INT UNSIGNED	

alert		
aid	INT UNSIGNED	<pk>
sid	INT UNSIGNED	<fk1>
sig_id	INT UNSIGNED	<fk2>
ip_src	INT UNSIGNED	
ip_dst	INT UNSIGNED	
ip_proto	INT UNSIGNED	
port_src	INT UNSIGNED	
port_dst	INT UNSIGNED	
start_time	DATETIME	
end_time	DATETIME	
priority	INT UNSIGNED	
count	INT UNSIGNED	

sensor		
sid	INT UNSIGNED	<pk>
name	varchar(30)	
system	varchar(30)	
ip	INT UNSIGNED	

icmphdr		
aid	INT UNSIGNED	<pk,fk>
icmp_type	TINYINT UNSIGNED	
icmp_code	TINYINT UNSIGNED	
icmp_csum	SMALLINT UNSIGNED	
icmp_id	SMALLINT UNSIGNED	
icmp_seq	SMALLINT UNSIGNED	
Column_7	<Undefined>	

data		
aid	INT UNSIGNED	<pk,fk>
data_payload	TEXT	

iphdr		
aid	INT UNSIGNED	<pk,fk>
ip_src	INT UNSIGNED	
ip_dst	INT UNSIGNED	
ip_ver	TINYINT UNSIGNED	
ip_hlen	TINYINT UNSIGNED	
ip_tos	TINYINT UNSIGNED	
ip_len	SMALLINT UNSIGNED	
ip_id	SMALLINT UNSIGNED	
ip_flags	TINYINT UNSIGNED	
ip_off	SMALLINT UNSIGNED	
ip_ttl	TINYINT UNSIGNED	
ip_proto	TINYINT UNSIGNED	
ip_csum	SMALLINT UNSIGNED	

tcphdr		
aid	INT UNSIGNED	<pk,fk>
tcp_sport	SMALLINT UNSIGNED	
tcp_dport	SMALLINT UNSIGNED	
tcp_seq	INT UNSIGNED	
tcp_ack	INT UNSIGNED	
tcp_off	TINYINT UNSIGNED	
tcp_res	TINYINT UNSIGNED	
tcp_flags	TINYINT UNSIGNED	
tcp_win	SMALLINT UNSIGNED	
tcp_csum	SMALLINT UNSIGNED	
tcp_urp	SMALLINT UNSIGNED	

udphdr		
aid	INT UNSIGNED	<pk,fk>
udp_sport	SMALLINT UNSIGNED	
udp_dport	SMALLINT UNSIGNED	
udp_len	SMALLINT UNSIGNED	
udp_csum	SMALLINT UNSIGNED	

Figure 3.4: The schema of Alert Warehouse.

Chapter 4: Data Preprocessing Phase

Since organizations of present subnets become more and more complex, it is common for a local server to take over hundreds of hosts and to collect thousands of alerts form these hosts, and these numerous alerts make alert analysis more difficult. Besides, the most part of present attacks are target-specific and stealthy intrusions instead of large-scale violence. Therefore, it is helpful for administrators to focus on some specific unknown attacks if we can use the previous system profile database to provide administrators some commands by looking for specific sensors with particular conditions. To discover those unknown and target-specific intrusions is the reason why we think that sensor grouping is very important. Besides, this idea is easy to be extended to the scale of one whole subnet or a single sensor.

Moreover, there are large amounts of detailed alert transactions in alert warehouse, so it is necessary to select some appropriate attributes and do format transformation according to specific analysis methods before execution of analysis procedures. Administrators may have different choices to get specific results with different data formats because of individual experiences and knowledge. Our idea is to provide some rules collocated with the format transformation algorithm according to different analysis methods, it is used to assist administrators to make decisions of appropriate data formats and data dimensions. Finally, data sources of raw alerts in one selected sensor group are transformed into specific data sources of Alert Filtering Phase.

In this chapter, Sensor Grouping (SG) algorithm is proposed to assist administrators selecting appropriate conditions of target sensor groups. Moreover, Alert Format Transformation (AFT) algorithm is proposed to execute data format transformation selected by administrators.

4.1 The Meta Knowledge of Grouping Sensors

Before discussing about how to divide sensors in a subnet into several objective groups according to their system characteristics, it is necessary to discuss the issues of that: how do attackers decide their targets?

(1) Attacks to single host

Most of attacks in this situation are that attackers have known intimately about environments of attack targets, and they can use the information such as system vulnerabilities and IP address to execute intrusions, or victims download insecure programs actively to be attacked. For an example, virus is a kind of attack in this situation.

(2) Attacks to several hosts with specific conditions

Attacks in this situation usually aim at hosts with specific conditions such as running a special OS or providing particular services, or probe hosts in a section of objective network address. This is most common attack situation because it is easy to get necessary information of victims stealthily and efficiently. For examples, Rootkits and Worms are this kinds of attacks.

(3) Attacks to a large amount of hosts

Attacks in this situation execute intrusions to many hosts without specific targets. Its purpose causes an obstruction on whole networks to prevent hosts from normal executions. For an example, DDoS is a kind of this situation.

It is necessary to pick up appropriate objective sensor groups as data sources according to required attack types for administrators. With appropriate sensor groups as targets, specific attack patterns could be discovered more precisely; on the contrary, it is hard to confirm specific attack patterns with improper sensor groups because of surplus noise alert data. According to all situations as given above, we conclude relations between intrusions and specific conditions as rules in the following table after surveying many security threat reports. These rules are used to provide administrators some proposed attributes of system profile database, and then administrators adjust these attributes by adding new attributes or pruning some attributes and set the values of selected attributes. After querying system profile database, we can get one or more sensor groups, and the subsequent filtering and analysis procedure is executed aiming at these groups.

RuleClass for Sensor Grouping algorithm

```
Rule 4.1.1    IF ( Target Attack = ‘Rootkit’ )
              THEN Select Attributes = ( OS, Service, Port )
Rule 4.1.2    IF ( Target Attack = ‘DDoS’ )
              THEN Select Attributes = ( Alarm Frequency = High,
              Download_Bandwidth = High )
Rule 4.1.3    IF ( Target Attack = ‘Worm’ )
              THEN Select Attributes = ( OS, Version, Anit-Virus = None )
Rule 4.1.4    IF ( Target Attack = ‘Inner Attacker’ )
              THEN Select Attributes = ( Anti-Virus = None,
              Download_Bandwidth = Low, Up_Bandwidth = High )
Rule 4.1.5    IF ( Target Attack = ‘Virus’ )
              THEN Select Attributes = ( OS, Anti-Virus = None )
```

This table is used to provide administrators proposed attributes, and it consists of some rules. These rules are designed corresponding to specific characteristics of attacks, and administrators can modify these rules dynamically.

4.2 The Heuristic of Grouping Sensors

Using the rules as above, a Sensor Grouping (SG) algorithm shown in Algorithm 4.1 is proposed to interact with administrators to assist them constructing specific sensor groups with appropriate attributes. We provide some proposed attributes according to specific attacks as references, and administrators can make a decision if these attributes are enough for grouping or not. If not, system will provide more information about other attributes for administrators and highlight some outstanding ones of these for more suggestions. Moreover, it is more flexible for administrators to do sensor grouping or not according to their requirements, or just select a single host for the following filtering procedure.

```
Input:    Database of System Profile
Output:   Sensor Behavior Groups (SBG)

Step 1:   Set SensorID as the main output attribute.
Step 2:   Ask experts for choosing a specific target attack type.
Step 3:   Use this specific attack type to determine the selected attributes for querying database of system profile by Rule 4.1.x.
Step 4:   Ask experts if these attributes are enough for dividing all sensors into groups.
          Step 4.1: List all kinds of values and their frequencies in each surplus attributes and highlight the specific values with quite different frequencies for administrators as references of attributes.
          Step 4.2:  IF NOT, ask experts for selecting one more appropriate attribute; GOTO Step 4.
          Step 4.3:  IF YES, GOTO Step 5.
Step 5:   Ask experts if there is any unnecessary attribute for dividing some sensors into a group.
          Step 5.1: List all kinds of values and their frequencies in each selected attributes and highlight the specific values with quite different frequencies for administrators as references of attributes.
          Step 5.2:  IF YES, ask experts for selecting one unnecessary attribute; GOTO Step 5.
          Step 5.3:  IF NOT, GOTO Step 6
Step 6:   Ask experts if they need to set the specific values of all selected attributes according to the requirements of experts.
Step 7:   Query database of system profile by upper selections and values, the querying result are Sensor Behavior Groups with specific attributes.
Step 8:   Ask experts if these groups are good enough to fit in with the requirements.
          Step 8.1   IF NOT, GOTO Step 2.
          Step 8.2 IF YES, GOTO Step 9.
Step 9:   Store all SBGs with their own unique SBG number.
Step 10:  Output all SBGs with their own unique SBG number.
```

Algorithm 4.1:   Sensor Grouping (SG) Algorithm

4.3 The Format Transformation of Alerts

After making decisions of target sensor groups, we can do alert format transformations to direct at these specific groups. According to different analysis methods, there are several corresponding data formats more appropriate than others for alert analysis, even there are many different formats or data dimensions for one specific analysis method to execute analysis. That is why administrators must select appropriate data format according to their requirements before executing selected analysis algorithms. In this stage, many kinds of data formats which are transformed from alerts of alert warehouse are proposed. At first, system provides some aggregated information for administrators as references such as diagrams of curves to show obvious suggestions, and that will assist administrators selecting an appropriate main analysis dimension of concept hierarchies. After that, the rules shown below are proposed to assist administrators to make decisions of data formats for subsequent alert analyses.

RuleClass for Alert Format Transfomation (AFT) algorithm.

Rule 4.3.1 IF (main analysis dimension = Time)
THEN target format selections = (alarm sequence, alarm transaction with specific attributes, alarm transaction with complete attributes)

Rule 4.3.2 IF (main analysis dimension = IP)
THEN target format selections = (alarm transaction with specific attributes, alarm transaction with complete attributes)

Rule 4.3.3 IF (main analysis dimension = PORT)
THEN target format selections = (alarm transaction with specific attributes, alarm transaction with complete attributes)

Rule 4.3.4 IF (main analysis dimension = None)
THEN target format selections = (alarm transaction with specific attributes, alarm transaction with complete attributes)

In this part, the goal of rules is corresponding to some special design of concept hierarchies in alert warehouse. After administrators selecting a main dimension, some possible data format selections are provided according to these rules for administrators to decide format policies. The meanings of main dimensions are expected permutation and transformation of alerts in alert analysis procedures. ‘Time’ main dimension represents the event time of security alerts; ‘IP’ main dimension represents the IP address of the source host; ‘PORT’ main dimension represents the target port of the destination sensor; ‘None’ main dimension represents that administrators feel like to select appropriate attributes in raw alert warehouse without selecting a main dimension.

After selecting a target data format, it is necessary for administrators to select the concept level, the start point and the end point of alert transactions because that different concept levels of each data format makes different results. There is a advantage for using the idea of concept hierarchies because integrated alert transactions can be analyzed on multiple dimensions and different concept levels in alert cube using operations such as roll-up, drill-down, slicing, etc.

For a example, after selecting ‘IP’ as the main analysis dimension and ‘alert transaction’ as target alert format with some specific attributes, administrators still need to select appropriate concept levels between ‘IPAddr’, ‘classC’, ‘classB’ and ‘classA’, and then set the start IP address and the end IP address of the target source hosts.

An Alert Format Transformation (AFT) algorithm shown in Algorithm 4.2 is proposed to interact with administrators to assist them making policies of the best appropriate alert format.

```
Input:    Raw data of IDS Alarm Warehouse
          Sensor Behavior Group (SBG)
Output:   Alarm Transactions with specific format in one SBG

Step 1:   List all relation diagrams of curves between alarm frequencies and
          TIME dimension, alarm frequencies and IP dimension, alarm
          frequencies and PORT dimension. Hightlight quite different values in
          these diagrams of cureves with different concept hierarchies of
          dimensions for administrators as references of selecting a main analysis
          dimension.
Step 2:   Ask experts for choosing a main analysis dimension with Rule 4.3.x
          according to the requirement of administrators.
Step 3:   Provide the corresponding target format selections to experts
Step 4:   Ask experts for choosing a specific target format.
Step 5:   Ask experts if there is any necessary appending attributes
          Step 4.1:  IF YES, make a attribute list for experts to make choices.
          Step 4.2:  IF NO, GOTO Step 5.
Step 6:   Ask experts for choosing a appropriate concept hierarchy according to
          main analysis dimension.
Step 7:   Ask experts to set the start point and the end point of main analysis
          dimension.
Step 8:   DO transformation of resource data of IDS Alarm Warehouse in one
          specific SBG into Alarm Transactions with specific format
Step 9:   Output these Alarm Transactions with specific format in one SBG.
```

Algorithm 4.2: Alert Format Transformation (AFT) Algorithm

4.4 Example for Data Preprocessing

According to the above-mentioned algorithms, we design corresponding concrete query methods and a specific alert transaction format to illustrate procedures and concepts in this phase.

In SG algorithm, system provides suggested query policies according to the requirement of administrators, and administrators can alter theses default attributes at will. As an example of discovering ‘Rootkits’ attack type, if administrators use the default suggestion of attributes, system generate a query policy as following automatically in SG algorithm.

Database of System and Network Profile Query Criteria for Rootkits

```
SELECT      SensorID, OS, Service, Port
FROM        All_Alarms
GROUP BY    OS, Service, Port
ORDER BY    SensorID
```

Figure 4.1: An example of query policy for Rootkits in database of system and network profile.

After that, administrators can get a sensor grouping result according to characteristics of OS, Service and Port. That may generate lots of sensor groups, and administrators can choose specific one of these or execute analysis procedure one by one.

The next step in this phase is to transform alerts into specific data format according to requirements of administrators after choosing a target sensor group. Continuing with the above example, if administrators choose a sensor group which all of sensors in it provide FTP service on port 21 with Windows Server 2003 operating system, and administrators have an idea of analysis alerts on sequential relations in this sensor group. Therefore, administrators choose ‘Time’ as the main analysis dimension, ‘alert sequence’ as specific data format and ‘Hour’ as unit of concept hierarchy. Besides, they must set the start point and the end point of event time, and alerts triggered in this range are extracted as target alert sources. The demonstration of alert data cube query concept in this example is as following.

```
SELECT      sig_name AS Signature, ip_dst
FROM        All_alert
WHERE       ( ip_dst BETWEEN clusterSID_Start AND clusterSID_End )
            AND ( [timestamp] BETWEEN time_slice_start AND
            time_slice_end )
GROUP BY    ip_dst
ORDER BY    [timestamp]
```

Figure 4.2: An example of query policy in data cube of alert warehouse.

After transformation of alert format, an integrated vector format is generated as following. The alert transactions in this thesis are named as Alert Sequences (AS) to differentiate from other alert formats of different methods.

System and Network Profile of selected SBG			
Group ID	OS	SERVICE	OpenPort
	WIN Server 2003	FTP	21
Alarm Sequence Transactions with format = ‘alarm sequence’			
SensorID	Time Slice 1	Time Slice 2	……
H1	X,F,F,E,X,G,A	G,E,V,T,Y,A	……
H2	Y,A,D,B,E	G,H,A,E	……

Note 1: Each character in alert sequences represents a specific alert signature name classified by IDS.

Note 2: ‘Hour’ is selected as the unit of concept hierarchy in this example, so the range in each time slice is 1 Hour/ per time slice.

Note 3: Different main analysis dimensions fit different data format vectors, but the same main analysis dimensions fit similar data format vectors with little different units of some attributes. Before provided to administrators, these detail specifications must be defined well first.

Chapter 5: Alert Filtering Phase

Because of the designing characteristics of IDS, there are huge amounts of false alerts collected from IDS sensors. To achieve an objective of highly detection rate without missing any intrusion, the design of IDS signature-based rules are asked to be as powerful as possible, but that makes IDS become more sensitive at the same time. Some characteristics of intrusions are similar to those of normal behaviors, so that some normal network behaviors are triggered as alerts, and those are what we call false alerts. Those false alerts as noises affect the results of real analysis procedures certainly, so it is necessary to filter those false alerts before executing analysis. Through there are some researches which skip filtering process, we believe that filtering of dirty alerts has two advantages: first, it will increase the accuracy of alert analysis; second, the complex of data execution will be deduced at the same time.

There are some researches discussing about how to filter false alerts efficiently, and different data characteristics and different filtering heuristics brings quite different filtering results. Generally speaking, most of these researches use special analysis methods or compile expert experiences to construct filter models, and use this filter model to discard those highly-possible false alerts to get clearer data.

In this phase, a filtering procedure by integrating most concepts of this kind of researches is proposed to transform many filtering algorithms into corresponding filter methods in this phase. The collection of these filtering methods is like a big filtering algorithm library. These filtering methods are provided by using Alert Filtering Method Selection (AFMS) algorithm for administrators to choose appropriate filtering policies, and then clear alert transactions are obtained.

5.1 The Heuristic of Generating Filter Model

Before discussing the design of procedure, we must consider the issue of data characteristics first:

(1) Alert Frequency

According to different importance and bandwidth of hosts, their numbers of triggered alerts are very different obviously. Besides, the scale of subnets aggravate the difference of alert frequencies; the bigger scale a subnet is, the more total alert number is in that subnet. It is common to collect thousands of alerts in a busy subnet.

Every alert is seen as a basic unit in constructing filter models in AFMS algorithm. There are two advantages by doing this: first, it is easy to transform raw alert transactions to specific formats for each method of filter model constructions; second, it is common to differentiate false alerts form others, and the number of alerts decreases directly after being filtered.

(2) Alert Characteristic

There are full of false alerts in alert warehouse. According to the results of most researches, it is indicated that false alert rates of different IDSs are lain in between 60% to 90%. The most important concept of all, some researches indicate that some of these false alerts occur with similar patterns in the same network, such as specific alert sequences or frequent source IP addresses, and those normal behaviors are triggered as alerts but they are not intrusions in fact. In other words, the idea of these researches is that if we can discover frequent behavior patterns of alerts, these frequent patterns are mostly like to be false alerts.

The idea of AFMS algorithm is to construct one or more filter models to execute filtering by comparing all alerts with filter models. Those filter models may be constricted according to experiences of administrators, or some algorithms corresponding to alert characteristics to construct filter models, such as Data Mining or Neural Networks, etc.

(3) Attack Characteristic

There are also some characteristics in attacks, so it is possible for specific filtering methods to be more efficient to specific intrusions. For an example, some specific Rootkits will give rise to constant alert sequences, so it is more appropriate for these intrusions to use Sequential Pattern Mining to filter false alerts. For another example, worm is a kind of variable intrusions, so using Generic Algorithm to filter false alerts is better than others.

The following rules are proposed corresponding to several intrusions for administrators to select appropriate filtering methods.

RuleClass for Alert Filtering Method Selection (AFMS) algorithm.

Rule 5.1.1 IF (Target Attack = 'Rootkit' or 'Worm')
THEN suggested methods = (Sequential Pattern Mining,
Neural Network, Generic Algorithm)

Rule 5.1.2 IF (Target Attack = 'DDoS' or 'Virus' or 'Inner Attacker')
THEN suggested methods = (Classification Mining,
Association Rule Mining, Manual)

5.2 The Method for Alert Filtering

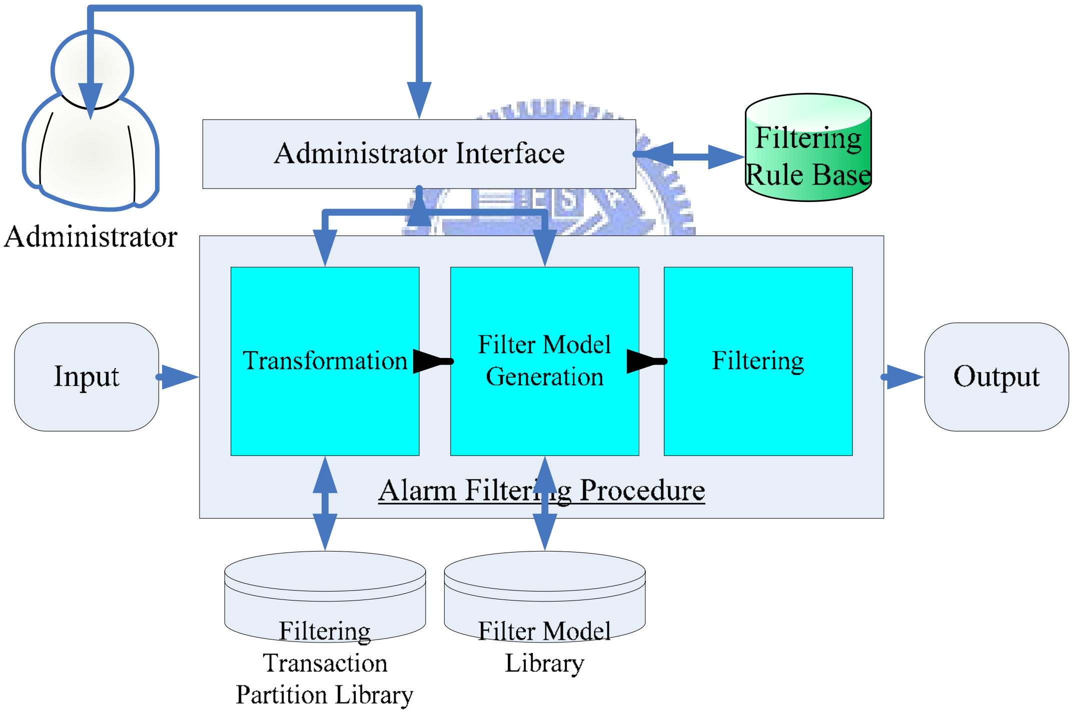


Figure 5.1: Flow Chart of Alert Filtering Procedure.

According to the rules shown in Section 5.1, administrators can select specific intrusion analysis target, and then some suggested filtering transformation partition policies and filter model policies are proposed to interact with administrators to make decisions of appropriate methods. In a complete alert filtering procedure, input alert transactions must be transformed into specific alert formats corresponding to the filter model, and then use these alert formats to construct filter model by selected algorithm. At last, the filter model is used to filter specific alerts by comparison. The flow chart of alert filtering procedure is shown in Figure 5.1.

AFMS algorithm shown in Algorithm 5.1 is used to interact with administrators to decide an appropriate combination of alert format and filtering method. The output data transaction formats must be the same with the input data formats but some false alerts are reduced.

```
Input:    Alarm Transactions with specific format in one SBG
Output:   Candidate Transactions in one SBG

Step 1:   Ask experts for choosing a specific target attack type.
Step 2:   Use this specific attack type to determine the suggested filter model
          construction methods by Rule 5.1.x
Step 3:   Randomly select a part of alarm transactions as a temporary set of alarm
          trasactions. DO every suggested alarm filtering algorithm with this
          temporary alarm set by the default setting of each algorithm. Show the
          filtering rate of each algorithm as references of selecting approapriate
          methods.
Step 3:   Ask experts for choosing an appropriate method to build the Filter
          Model(s).
Step 4:   IF it is necessary to do format transformation of each sensor in this
          method,
          DO transformation.
Step 5:   Generate the Filter Model(s) of the SBG by this method.
Step 6:   Compare all original alarm transactions with Filter Model(s) to filter
          false alarms.
Step 7:   Generate the Candidate Transactions in one SBG.
```

Algorithm 5.1: Alert Filtering Method Selection (AFMS) Algorithm

5.3 Example for Alert Filtering

According to the above alert filtering procedure, we design a filtering algorithm corresponding to discover attack sequences of Rootkits. This algorithm consists of three phases of Transfomation, Filter Model Generations and Filtering, too. At first, alert format transformation must be executed. According to alert sequence data type of Rootkits, we have a heuristic that two repeat alerts in one alert sequence are meaningless, and it is because that most of Rootkits alert sequences have no repeat alerts during attacking. We use this heuristic to execute alert transaction partitions. Besides, we have a basic idea: frequent behavior, over an extended period of time, is likely to be normal. In the other words, a modified sequential pattern mining method AprioriAll [2] is used to discover frequent sequences of alerts in single sensor, and these frequent sequences are seen as false alert patterns and collected as a filter model. At last, the filter model is used to reduce false alerts and clear data of alerts are aggregated to initial input data formats. The special algorithm proposed by us is shown in Algorithm 5.2. To fit in with requirements of flexibility and robustness for administrators in such a knowledge-based approach, system interacts with administrators to decide sub-algorithm of partition policies and the value of minimum support in AprioriAll algorithm dynamically. That makes it possible for administrators to make different decisions according to different situations.

```
Input:    Alarm Sequences (AS) with specific format in one SBG
Output:   Single-Sensor Candidate Sequences of all sensors in one SBG

Step 1:   For each sensor i, i=0~m-1, m is the number of all sensors in this SBG.
Step 2:       Ask administrators to choose a specific Partition Algorithm.
Step 3:       Divide all alarm sequences of i with this Partition Algorithm as
              Alarm Sub-Sequence Transactions (ASSTs).
Step 4:       Ask Administrators to decide some appropriate values of minimum
              support in AprioriAll. (or system generated automatically.)
Step 5:       For each different value of minimum support in AprioriAll
              Step 5.1:  Generate frequent sequences of i by AprioriAll with one
              specific value of minimum support.
              Step 5.2:  Store all frequent sequences as the FM[i].
              Step 5.3:  Filter all ASSTs of i by compared with FM[i] and store
              the results after being filtered.
              Step 5.4:  Claculate the filtering rate with this specific value of
              minimum support.
              Step 5.4:  IF there are still values of minimum support without
              being tested, GOTO Step 5.
              ELSE GOTO Step 6.
Step 6:       Ask administrators to choose an appropriate set of results with one
              specific value of minimum support.
Step 8:       Combine all remaining sequences as a new Single-Sensor,
              Candidate Sequences (SSCS) of   sensor i.
Step 9:    Store and output SSCSs of all sensors in one SBG.
```

Algorithm 5.2: A special example of Alert Sequence Filtering (ASF) Algorithm.

There are varied partition policies in above algorithm for administrators to execute format transformation. We list three partition policies as follows.

```
Partition Algorithms:

CASE 1 (First Non-Repeat Policy)
    Step 1:    Scan every alarm from the first to the end;
    Step 2:        If there is an equal alarm in front of the present alarm
    Step 3:            Do partition from the first element to the former element
                       of the present element;
    Step 4:    The rest is a new alarm sequence and Do scan again until every
               alarm is scanned.

CASE 2 (Last Non-Repeat Policy)
    Step 1:    Scan every alarm from the end to the first;
    Step 2:        If there is an equal alarm in front of the present alarm
    Step 3:            Do partition from the last element to the next element of
                       the present element;
    Step 4:    The rest is a new alarm sequence and Do scan again until every
               alarm is scanned.

CASE 3 ( Equi-Length Policy)
    Step 1:    Ask Administrators to set a value of the sequence length.
    Step 2:    Partition each alarm sequence into several subsequences by the
               fixed length.
```

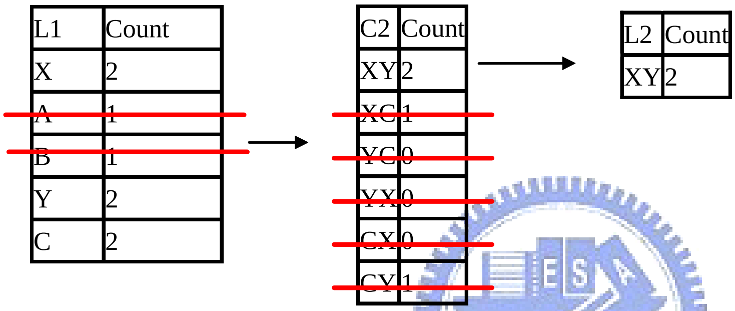
Here we use an example to illustrate our whole specific algorithm. We suppose that there is an alert sequence of sensor H1 in time slice T1 as following:

SensorID	OS	SERVICE	OpenPort	T1
H1	XP	HTTP	80 …	XABYXCYC

We use ‘First Non-Repeat Policy’ as our partition policy for example. Let AS[7] be alert sequence: XABYXCY; because AS[4] equals AS[0], so this sequence is divided into two alert sequences: XABY and XCYC, and executes partition again in the rest as XCYC which becomes new AS[4]. In AS[4], we can find that AS[4] equals AS[2] again, so this sequence AS[4] is divided into two alert sequences: XCY and C. After executing partition in whole alert sequence, we can get three new alert sub-sequence transactions (ASST) such as XABY, XCY and C. The new format after transformation is as follows.

SensorID	OS	SERVICE	OpenPort	T1		
H1	XP	HTTP	80 …	XABY	XCY	C

Then the step of Filter Model Generation is executed, and the filter model FM is constructed by AprioriAll algorithm. XABY, XCY and C become three input data transactions of AprioriAll, and the value of minimum support is supposed to be 2.



L1	Count
X	2
~~A~~	~~1~~
~~B~~	~~1~~
Y	2
C	2

C2	Count
XY	2
~~XC~~	~~1~~
~~YC~~	~~0~~
~~YX~~	~~0~~
~~CX~~	~~0~~
~~CY~~	~~1~~

L2	Count
XY	2

Therefore, we can get a frequent sequence as XY. All frequent sequences in one single sensor are collected as a filter model FM of this sensor, so FM of sensor H1 is XY.

Next filtering operation is executed. Comparing all ASSTs with elements in this FM, if there exists one subsequence (continuous or discontinuous) of ASSTs equals some element in the FM, discard this subsequence in the ASSTs. In this example, because there are XABY and XCY of ASSTs including an element of FM as XY, the subsequence XY is discarded in all ASSTs. Finally, new ASSTs become AB, C and C. These filtered ASSTs must be integrated to a clear alert sequence in the same format with the input data format, so the following output alert sequence is ABCC named Single-Senor Candidate Sequence (SSCS) in this thesis.

SensorID	OS	SERVICE	OpenPort	T1
H1	XP	HTTP	80 …	ABCC

Chapter 6: Collaborative Analysis Phase

In Data Preprocessing Phase, the idea of target sensor grouping is proposed according to requirements of administrators to aim at specific targets and increase accuracy of analysis. In Alert Filtering Phase, our objective is to filter source alerts as clear as possible. Through these two phases, it is common to see those alerts as reliable sources for pattern analyzing. To find specific patterns in these kinds of numerous sources is likely to discover meaningful patterns in distributed databases or data warehouses for decision support. There are many researches discussing how to mine behavior patterns in databases, and different analysis methods with different data sources cause different outcomes. In our thought, single analysis algorithms are not enough powerful for highly-correct intrusion detections. If administrators could be provided with many analysis policies according to requirements of them to make decisions, these flexible analysis policies would make analysis procedures more effective than single analysis method.

Besides, different analysis results may be discovered according to different sensor groups, and some relations of results between these different sensor groups are very likely to be meaningful for administrators to conclude overall information to illustrate and solve the problems caused by intrusions. In our thought, it is important to design methods of information sharing to integrate and exchange specific analysis results, and these methods will reduce the security load of administrators.

In this Phase, an Intra-Group Collaborative Analysis Selection (IGC-A) algorithm is proposed to assist administrators selecting appropriate analysis algorithms according to requirements of them, and then the system analyzes alerts and outputs results for administrators to fix root causes. It results in administrators focusing on the most important and valuable core tasks. These analysis algorithms stored in analysis method library of the system can be easily to added or modify according to evolutions of new knowledge about unknown intrusions. Moreover, an Inter-Group Collaborative Behavior Sharing (IGC-B) algorithm is proposed to provide specific output formats corresponding to different analysis methods, and then these formats are used to execute information integrating and sharing to promote efficiencies of administrators.

6.1 The Collaborative Concept

Before illustrating collaborative analysis procedures, it is necessary to discuss about the issues of collaborative defense on intrusion detection.

(1) Definitions

We redefine the definition of collaborative defense on intrusion detection. In most traditional IDSs, the definition of collaborative defense methods is to design a system architecture and some special data formats, and those are used to make information sharing more quickly between administrators of different organizations. With these environments, messages are sent to notify attack situations or ask other experts for assistance, as ancient false fires. Because our system is constructed in this kind of environments, it is common to classify our research as a kind of collaborative defense researches. Besides, a concept of grouping sensors with similar system profiles to co-analyze between many sensors is proposed in this thesis. It is common to extend this concept to information exchanging and sharing between different organizations, so this idea seems to be another type of collaborative defense. According to our new definition, it is necessary to discuss the following two situations.

(2) Multi-sensors information analysis in collaborative defense groups

The meaning of intra-group analysis is to discover suspicious intrusions which focus on specific conditions of hosts. In our thesis, the mentioned multi-sensors analysis is the same as our intra-group collaborative analysis method. Some specific methods are proposed to find suspicious patterns by comparisons of alerts between all sensors in a specific sensor group. Because there are some special similar system and network profiles of sensors in one group, it is feasible to aggregate analysis results and these profiles to be provided as references for administrators.

IGC-A algorithm is used to interact with administrators to select appropriate analysis methods, execute output results and integrate results with specific profiles into pre-defined information formats for final outcomes.

(3) Multi-groups information aggregation between collaborative defense groups

The meaning of inter-group analysis is to enhance concept hierarchies of intrusion detection. There are some intrusions which do not focus on specific system profiles for intrusions, so it must cause information deficiencies if we just execute analysis on specific sensors. Our idea is to aggregate analysis results of the same analysis method between different sensor groups, and that provides administrators integrated information to make high-level decisions. IGC-B algorithm is used to describe this procedure of integration, and this idea is common to extend cross-organization information sharing in the Internet.

6.2 Intra-Group Collaborative Heuristic

In this section, pure alerts after filtering (through there are still some false alerts) are used to execute true analysis. There are many researches providing ideas of calculating and analyzing numerous data sources. Administrators can increase domain knowledge into analysis methods according to the characteristics of target intrusions. These analysis methods are likely to be signature rules written by experts, special scoring formulas corresponding to intrusion characteristics, data mining approaches of intrusion detection, or simple statistic tables on specific dimensions. The results in this step may have different output format because of different requirements, so definitions of output formats are a part of analysis methods. The flow chart of collaborative intra-group analysis procedure shown in Figure 6.1 includes three parts, and this flow chart conforms to most part of existing analysis algorithms. It is easy for administrators to design a new analysis algorithm or modify a traditional one according to this flow chart.

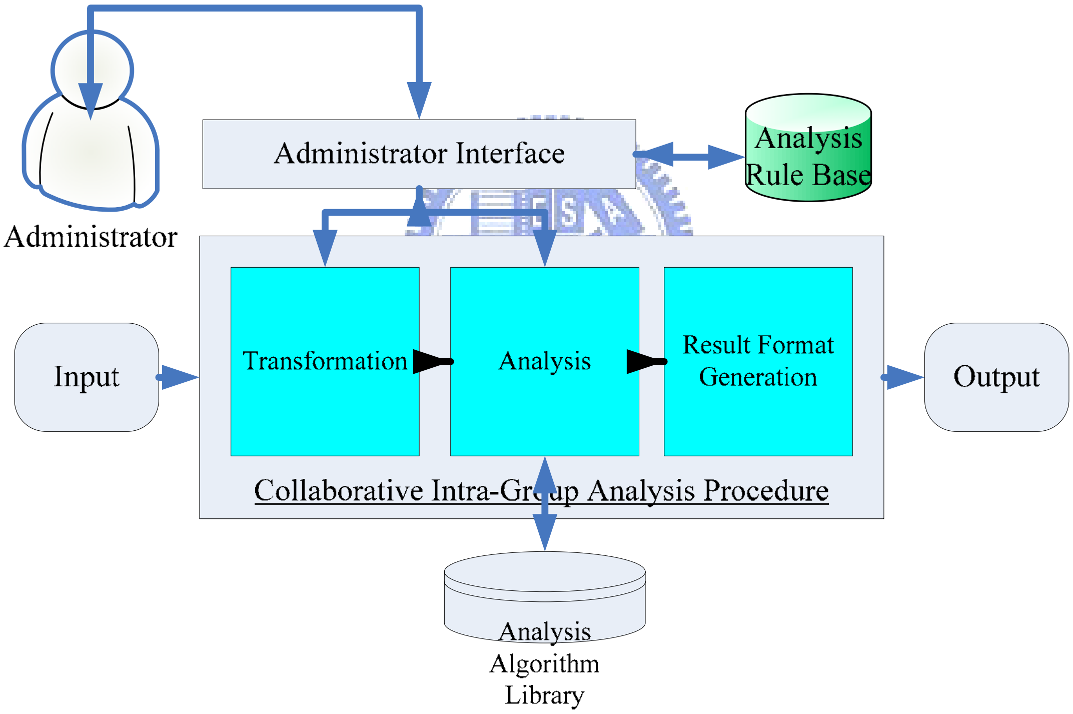


Figure 6.1: Flow Chart of Collaborative Intra-Group Analysis Procedure.

In the other words, system provides numerous analysis algorithms as a library, and interacts with administrators to decide appropriate analysis methods. Administrators can focus on one specific sensor group with different analysis methods for consulting analysis results to promote the accuracy of analyses. IGC-A algorithm is shown in Algorithm 6.1.

```
Input:   Candidate Transactions in one SBG
Output:  Suspicious Alarm patterns of selected analysis methods with their specific formats

Step 1:  Ask experts for choosing an appropriate Collaborative Intra-Group Analysis method according to the requirement of administrators.
         Step 1.1:  Randomly select a subset of all data inputs as a temporary data set to test performances of each analysis method.
         Step 1.2:  Execute each analysis methods with this temporary data set and show the results and detailed information for administrators.
         Step 1.3:  It is easy for administrators to make decisions of appropriate analysis methods by this information.
Step 2:  IF it is necessary for experts to set some values of specific variables in this method,
         Do the guidance for administrators to set these values.
Step 3:  IF it is necessary to do some alarm transformations to specific format in this method,
         DO the transformation process.
Step 4:  Run the selected analysis method.
Step 5:  Generate and store the results with the specific format in this method.
Step 6:  Ask administrators if these results of analysis methods are enough for security experts or not.
         Step 4.1   IF NOT, GOTO Step 1.
         Step 4.2   IF YES, GOTO Step 7.
Step 7:  Output all results of all the selected analysis methods with their specific formats in one SBG.
```

Algorithm 6.1: Intra-Group Collaborative Analysis Selection (IGC-A) Algorithm

6.3 Inter-Group Collaborative Heuristic

In one specific sensor group, we can use many analysis methods for co-analyzing mentioned in Section 6.2 to promote the accuracy; with similar concept, an idea of aggregating many results of the same analysis method between different sensor groups is proposed as references for administrators. The motivation of this section is to discover those widespread intrusions such as DDoS. Generally speaking, this operation raises the original concept hierarchies to analyze suspicious intrusion behaviors completely. The advantage of this concept is not only for administrators of single subnets but also for administrators between different organizations to exchange their knowledge of analysis results and discover more overall intrusion patterns in global view.

It is necessary to define appropriate alert exchange formats on alert exchanging of IDS. Extensible Markup Language (XML) which has been exercised popularly is used as our alert exchange format description language in this thesis, and there are many researches discussing about how to design data exchange formats by using XML. Because there are pre-defined data formats corresponding to every analysis method in IGC-A algorithm, it is common to design specific XML-based data exchange format of each analysis method; in this thesis, this kind of XML-based data exchange format is named as suspicious behavior description Markup Language (SBDML). Each result of analysis methods is mapping to a specialized SBDML, and this SBDML must cover all detailed information such as the version of analysis methods, suspicious alert patterns, and specific system and network profiles of selected sensor group. Generally speaking, this SBDML is likely to use a vector data structure to record information which is discovered by IGC-A algorithm. For example, a specific SBDML corresponding to one analysis method is proposed and its abstract vector format is as below.

Analysis Method	Group ID	# of Hosts	Suspicious Pattern	Suspicious Flag	OS	Service	Port

Using these kinds of SBDML, it is easy for administrators to execute suspicious alert behavior information aggregations, or even overall cross-organization alert information exchanges. The only hypothesis in this thesis is that we only mention on the issue of information aggregation with the same format of SBDML, because different formats of SBDML are corresponding to different analysis algorithms. An Inter-Group Collaborative Behavior Sharing (IGC-B) algorithm is proposed to describe how to execute information aggregation between the same SBDML transactions of several different sensor groups to provide more organized information for administrators.

```
Input:   Suspicious Alarm Pattern Vectors with their specific SBDML formats in many SBGs
Output:  Aggregated Information of Suspicious Alarm Patterns with a specific analysis method.

Step 1:  Divide all Suspicious Alarm Pattern Vectors into several classes with the same ‘Analysis Method’ value.
Step 2:  For every class with different ‘Analysis Method’ values
         Step 2.1: Generate tables for all possible values of ‘Suspicious Pattern’ and ‘Suspicious Flag’.
         Step 2.2: In each table, list all possible values of System and Network Profile attributes in this specific SBDML format and calculate their occurrence rates; record these information in the table.
Step 3:  IF there exists any class which has not been transform into tables, THEN GOTO Step 2.
         ELSE the algorithm ends.
```

Algorithm 6.2: Inter-Group Collaborative Behavior Sharing (IGC-B) Algorithm

6.4 Example for Collaborative Analysis

According to the above analysis procedure, we propose an analysis algorithm in order to discover suspicious alert sequences of Rootkits in Intra-Group Collaborative Analysis Phase. This algorithm also includes three steps of Transformation, Analysis and Result Format Generation. At first, it is necessary to transform alert transactions into specific data formats. Because we have a thought of inspecting each possible alert sequence strictly, all input SSCSs are divided into 2-candidate alert subsequences (2-candidate means the length of this sequence is 2). Besides, two variables are designed to record frequencies and locations of each 2-candidate alert subsequences between several continuous time slices as references. Our idea of real analysis is to use specific scoring methods to model possible behaviors of Rootkits. According to the characteristics of Rootkits, higher the scoring value is, more suspicious the alert pattern is. Finally specific thresholds are set to flag some special situations as suspicious attack patterns, then administrators are noticed to trace the causes of suspicious patterns and fix intruded hosts. After referring to numerous researches of Rootkits, we conclude a table shown in Figure 6.2 about characteristics of Rootkits, so our scoring policies are necessary to catch these intrusions as good as possible. Two scoring policies are proposed for administrators to make decisions according to different situations. Moreover, it is common for administrators to modify or add scoring policies corresponding to their domain knowledge. With these scoring policies, some rules are proposed to determine 2-candidate alert sequences with specific flags if there is any 2-candidate alert sequence conforming to one of these rules. At last, analysis results are aggregated and then transformed to pre-defined output data format of SBDML to provide administrators making decisions on security issues.

Situations	Ratio	Analysis Difficulty	Analysis Methods
Attacks to single host	Low	Hard	--------
Attacks to many hosts in seconds	Medium	Medium	Statistics
Attacks to many hosts in many time slices	High	Hard	Variability Analysis

Figure 6.2: The table of characteristics on Rootkits

An Intra-Group Suspicious Alert Sequence Analysis (IGSASA) algorithm shown in Algorithm 6.3 is proposed as an example. In this algorithm, a variable ‘Score’ is used to represent the variation of a specific 2-candidate alert subsequence, and a variable ‘Repeat’ is used to represent the frequency of a specific 2-candidate alert subsequence with the same situations. Generally speaking, these two variables represent two contrary meanings in fact. Administrators have abilities to make decisions of scoring policies and flagging rules according to requirements of them.

```
Input:    Single-Sensor Candidate Sequences of all sensors in one SBG
Output:   Suspicious Alarm Sequences with their flags and the profile of this SBG

// Transformation phase
Step 1:   For each sensor j, j=1~n (n is the number of sensors ),
          Transform the SSCS of j into 2-candidate subsequences;
Step 2:   For each 2-candidate subsequence,
          Step 2.1:  Store the Hosts value;
          Step 2.2:  Calculate the Percentage value;
Step 3:   Store results of all 2-candidate subsequence;

//Scoring Phase
Step 4:   Ask administrators to choose a Scoring Policy to analysis all 2-candidate
          subsequence
Step 5:   Randomly select a subset of all data inputs as a temporary data set to test
          the maximum values of variables as references of value setting for
          administrators. (For example, set values of variables as 80% of the
          maximum values)
Step 6:   Ask administrators to set values of variables in this policy. ( Such as
          Threshold(score) and Threshold(repeat). )
Step 7:   Compare the values of Hosts and Percentage between T(i-1) and T(i) in
          the same 2-candidate subsequence to calculate the values of Score and
          Repeat by selected Scoring Policy. ( i=2~m, m is the number of time
          slices ).

//Flagging Phase
Step 8:   Check the values of Score and Repeat in each 2-candidate subsequence
          if there is any 2-candidate subsequence matching the special Flagging
          Rules.
          Step 7.1:  IF YES, trigger it with a flag of the special rule; GOTO Step
          7..
          Step 7.2:  IF NO, GOTO Step 8.
Step 9:   Aggregate the successional subsequences with the same suspicious flags
Step 10:  Output Suspicious Alarm Sequences with their flags and the profile of
          this SBG.
Step 11:  Transform these results into SBDML format records and store them as
          references.
```

Algorithm 6.3: An example as Intra-Group Suspicious Alert Sequence Analysis (IGSASA) algorithm.

Two scoring policies and an example set of flagging rules are proposed as following corresponding to characteristics of Rootkits.

Scoring Policies

```
CASE 1:  ( Formula-based )
         IF        ( Hosts(i)=Hosts(i-1) AND Percentage(i)= Percentage(i-1) )
         THEN      Set Score=0 AND Repeat++;
         IF        ( Hosts(i)!=Hosts(i-1) AND Percentage(i)= Percentage(i-1) )
         THEN      Set Score = Score + ½ [ |Sets(t)⊕Sets(t-1)| * (1/n) ] ( n is the
                   number of sensors in this Group )
                   AND Repeat=0;
         IF        ( Hosts(i)!=Hosts(i-1) AND Percentage(i)!= Percentage(i-1) )
         THEN      Set Score = Score + |Sets(t)⊕Sets(t-1)| * abs[P(t)-P(t-1)] AND
                   Repeat=0;

CASE 2: ( Frequency-based )
         IF        ( Hosts(i)=Hosts(i-1) AND Percentage(i)= Percentage(i-1) )
         THEN      Set Score-- AND Repeat++;
         IF        ( Hosts(i)!=Hosts(i-1) AND Percentage(i)= Percentage(i-1) )
         THEN      Set Score++ AND Repeat++;
         IF        ( Hosts(i)!=Hosts(i-1) AND Percentage(i)!= Percentage(i-1) )
         THEN      Set Score++ AND Repeat--;
```

Flagging Rules

```
    RULE 1: IF        Score > threshold(score)
            THEN      flag as “Highly Suspicious”;
    RULE 2: IF        (Repeat > threshold(repeat) & percentage!=0 )
            THEN      flag as “Temporal Frequent”;
    RULE 3: IF        percentage == 100%
            THEN      flag as “Spatial Frequent”;
    RULE 4: IF        else
            THEN      flag as “Unknown”;
```

We use an example to illustrate the above algorithm easily. The data shown as follows are SSCSs after filtering.

System Profile				SSCSs			
SensorID	OS	SERVICE	Port	T1	T2	T3	T4
H1	XP	HTTP	80	ABCC	Z	DE	F
H2	XP	HTTP	80	BC	DEF	Z	X
H3	XP	HTTP	80	AY	BC	F	DEF
H4	XP	HTTP	80		B		ABC

SSCSs of all sensors of Group 1 in time slice T1 are taken as examples for suspicious scoring as following.

	T1
H1	ABCC
H2	BC
H3	AY
H4	

2-candidate subsequences after format transformation are as following.

Group 1		
Host	Time	Subseq.
H1	T1	AB AC BC
H2	T1	BC
H3	T1	AY
H4	T1	

According to the above table, the variables “Hosts” and “Percentage” of each 2-candidate alert subsequences are calculated as following.

Group 1	Time 1	
Subseq.	**Hosts**	**%**
AB	H1	33
BC	H1 H2	66
AC	H1	33
AY	H3	33

All tables of results can be calculated in the same way as followings.

Group 1	Time 2	
Subseq.	**Hosts**	**%**
DE	H2	33
EF	H2	33
DF	H2	33
BC	H3	33

Group 1	Time 3	
Subseq.	**Hosts**	**%**
DE	H1	33

Group 1	Time 4	
Subseq.	**Hosts**	**%**
DE	H3	33
EF	H3	33
DF	H3	33
AB	H4	33
BC	H4	33
AC	H4	33

Finally, all above results are led in scoring phase of IGSASA and ‘Formula-based Scoring Policy’ is selected as the scoring policy in this example. We can get the result table as following. Here the values of threshold(score) and threshold(repeat) are supposed as 0.9 and 3 individually.

Group1	T1		T2		T3		T4	
	Score	Repeat	Score	Repeat	Score	Repeat	Score	Repeat
AB	0	0	.33	0	0	0	.33	0
BC	0	0	**.99**	0	--	--	--	--
AC	0	0	.33	0	0	0	.33	0
DE	--	--	0	0	.66	0	**.99**	0
EF	--	--	0	0	.66	0	**.99**	0
DF	--	--	0	0	.66	0	**.99**	0
AY	0	0	.33	0	0	0	--	--

Note: For this example, the Score of 2-candidate alert subsequence BC in the time slice T2 is higher than threshold(score), so BC is flagged as ’Highly Suspicious’ after T2; DE, EF, DF are flagged as ’Highly

Suspicious' after T4, and these three 2-candidate alert subsequences are successive, so they are aggregated as a bigger suspicious alert sequence as DEF with flag 'Highly Suspicious'.

Finally, the output results with specific formats of SBDML are generated. Here an example of output result format of abstract vector is proposed as following.

Analysis Method	Group ID	# of Hosts	Suspicious Pattern	Suspicious Flag	OS	Service	Port
IGSASA	Group 1	4	DEF	Highly Suspicious	XP	HTTP	80

Chapter 7: Case Study

In this thesis, the Knowledge-based framework Collaborative Discovering of Suspicious Network Behaviors as shown in Figure 3.1 is proposed to assist administrators discovering suspicious patterns of intrusions. In real environments of Internet, there may be hundreds of hosts in one intranet and thousands of alerts in one day. It is not easy to do system verification on such a large-scale intranet. To simulate real network environments as more as possible, we construct a small virtual intranet with one server and eight hosts for experiments. Before showing the experimental results, we first describe the experimental environments the dataset used.

There are two different roles in such an environment: IDS sensors and an IDS server. IDS sensors are used to trigger IDS alerts and store these alerts temporarily, and the load of sensors must be as less as possible to avoid interfering with common usages. IDS server is used to collect alerts as alert warehouse, maintain database of system and network profiles, and interact with administrators to execute discovering suspicious network behaviors.

The requirements of the experimental system include some related tools:

(1) IDS sensor: OS (FreeBSD, Linux, Windows), IDS sensor tools (Snort sensor), Database (MySQL).

(2) IDS Center for alert warehouse: OS (Windows Server 2003 IIS), Database (MS-SQL 2000 Server).

(3) Web-based Analysis Console: Web Server (Apache), PHP, BASE [24], Database (MySQL).

(4) Alert Analysis Console: Database Client (for MS-SQL 2000 Server), Analysis Service of MS-SQL 2000 Server, Expert System (DRAMA 2.6).

7.1 The Overview of The Related Tools

The Snort [27] is a signature-based intrusion detection system and open source software. It represents a cost-effective and robust NIDS solution that fits the needs of many organizations. The Snort is very flexible in the ways it can be deployed. Many security industry watchdogs include Snort signatures in their security announcements (such as CERT). When intrusions are ravaging the Internet and there are constantly new variants, even there are multiple updates weekly. The Snort mailing lists are fantastic resource for people who are trying to run Snort or write their own signatures. There are a number of applications that can act as central monitoring and alerting consoles, such as BASE [24].

The BASE is the Basic Analysis and Security Engine. It is based on the code from the Analysis Console for Intrusion Databases (ACID) project. This application provides a web front-end to query and analyze the alerts coming from a snort IDS system. The BASE is a web interface to perform analysis of intrusions that the Snort has detected on your network.

To post processing of alert transactions requires commercial databases. In this thesis, the MS-SQL is selected by us. The MS-SQL 2000 Server helps us to do data transformation services. It can automate processes to extract, transform and load data from heterogeneous sources. The MS-SQL 2000 Server Analysis Services includes OLAP, data mining and data warehouse tools. It makes better decisions, performs rapidly, and executes analysis on large and complex data sets using multi-dimensional storage.

The DRAMA [26] is applied for building up the decision support inference engine. DRAMA is a rule-based, client-server tool/environment for knowledge- based system development. It can assist knowledge engineers in building up an expert system or decision support system. Using the client-server architecture of DRAMA, the knowledge base or rule base is maintained on a server and clients could access this server for inference services.

7.2 The Environment Design

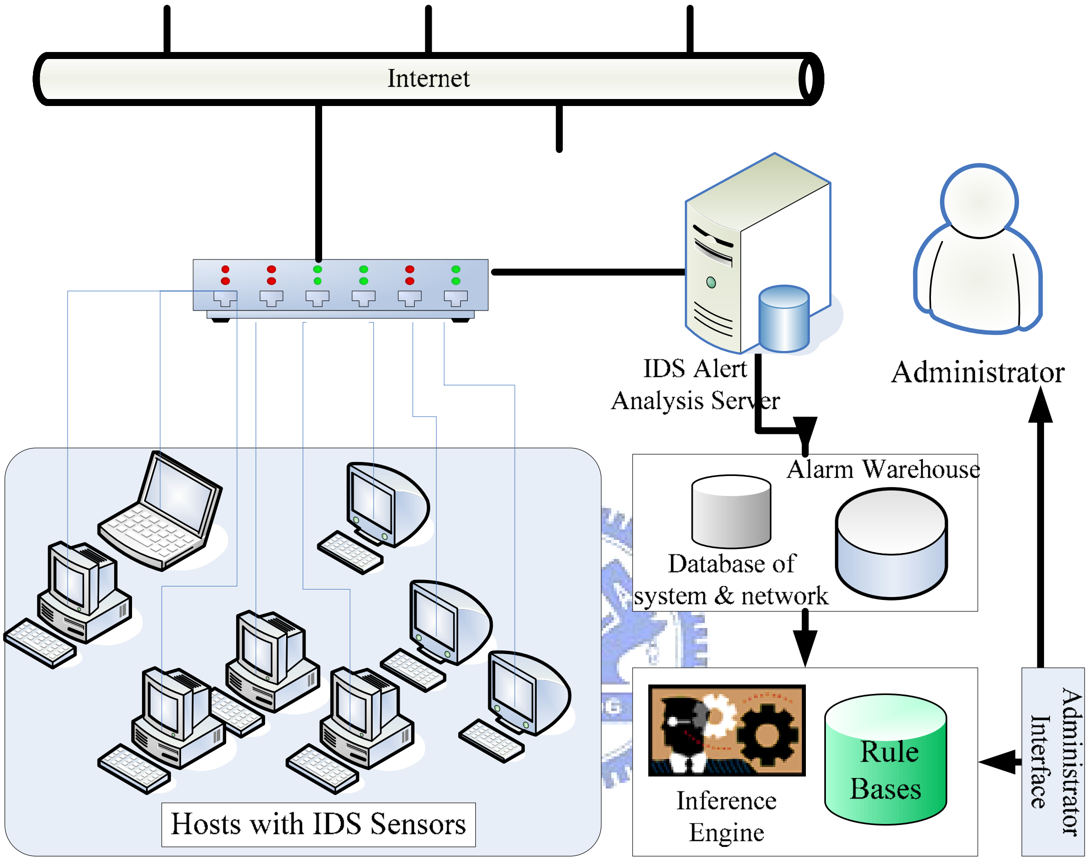


Figure 7.1: System prototype in experiments.

The knowledge-based architecture of collaborative discovering of suspicious network behaviors is implemented as shown in Figure7.1. All the related tools are described in Section 7.1; there is one server which plays the role of IDS Alert Analysis Server, including IDS center for alert warehouse, web-based analysis console and alert analysis console; besides, eight hosts all play the role of IDS sensors to trigger alerts as our data sources. The system and network profiles of these sensors are shown in Figure 7.2.

SID	IP	OS	Service	Port	Dow._ Band.	Up_ Band.	Anti- Virus	Alert _Freq.	……
H1	x.x.x.1	WinXP	HTTP	80	High	High	Y	High	……
H2	x.x.x.2	WinXP	HTTP	80	Low	Low	Y	High	……
H3	x.x.x.3	WinXP	HTTP	80	High	High	Y	High	……
H4	x.x.x.4	WinXP	HTTP	80	High	Low	Y	High	……
H5	x.x.x.5	Linux	FTP	21	Low	Low	N	Low	……
H6	x.x.x.6	Linux	FTP	21	Low	High	N	Low	……
H7	x.x.x.7	Linux	FTP	21	Low	High	N	High	……
H8	x.x.x.8	BSD	FTP	995	High	High	N	Low	……

Figure 7.2: The system and network profiles of all sensors.

We have conceptualized alerts according to the Snort rule set. The Snort is a network-based IDS where alerts are triggered by a collection of signature-based rules. Each Snort rule is composed of a Snort identification number, a message that is included in the alert when the rule is triggered, an attack signature, and references to sources of information about the attack. Each alert is provided with an identifier, time and data, sensor identifier, triggered signature, IP and TCP headers and payload. These alerts will be stored in the relational database as our alert warehouse. Alerts in one period of time (4 hours) are collected by IDS center as data source in this experiment, and the detailed contents of this alert transaction set are listed in Appendix. For easy reading, we replace the original alert signature names with different capital letters.

7.3 The Results

To verify the feasibility of our knowledge-based framework of collaborative discovering of suspicious network behaviors, we execute whole 3-phase analysis framework with alert transactions in Appendix, and Rootkits attacks are supposed as our target intrusion type.

In Data Preprocessing Phase, we use SG algorithm and AFT algorithm to execute alert data preprocessing. Because Rootkits is our target attack, the selected attributes in SG algorithm are OS, Service, and Port if we don’t modify other attributes. It is common to divide these sensors into 3 groups: the first group includes H1, H2, H3, and H4; the second group includes H5, H6, and H7; the last group includes only H8. Next we can construct alert sequence transactions by AFT algorithm shown in Section 4.4. “Time” is selected main dimension and “Hour” is the unit of concept hierarchy here.

In Alarm Filtering Phase, we use a special ASF algorithm with First Non-Repeat Policy shown in Section 5.3 to filter false alerts. To verify the performance of ASF algorithm, we compare alert reduction rate between our ASF algorithm and another existing filtering algorithm which has been discussed in [1]. The sequence length of filtering algorithm in [1] is fixed as 5, and the threshold is supposed as 0.4; in the other words, the value of minimum support in these two filtering algorithm are all 2. We use alert sequences of individual sensors as data sources of each algorithm. Figure 7.3 shows the result, and it shows that our algorithm is more stable and effective than the existing filtering algorithm of [1] in alert reduction rate.

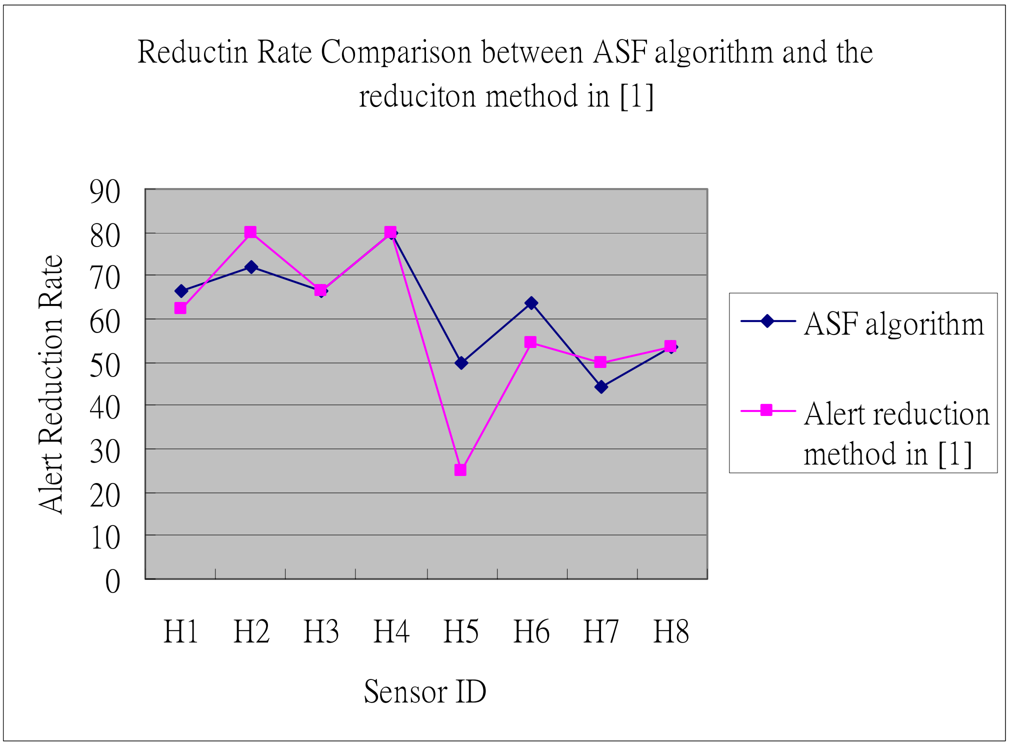


Figure 7.3: Exp1 - Comparison with existing filtering algorithm.

In Collaborative Analysis Phase, we use a special IGSASA algorithm shown in Section 6.4 to discover suspicious alert sequences. We use the "Formula-based" policy to calculate the values of Score and Repeat, and the values of threshold(score) and threshold(repeat) are supposed as 0.9 and 3 individually. Figure 7.4 shows the percentages of all 2-candidate alert subsequences with different suspicious flags in Group 1 and 2 (Group 3 is not discussed because there is only one sensor in it). The total value of percentages of each suspicious condition must not be too high or too low, or that will make it hard for administrators to generalize suspicious alert patterns efficiently.

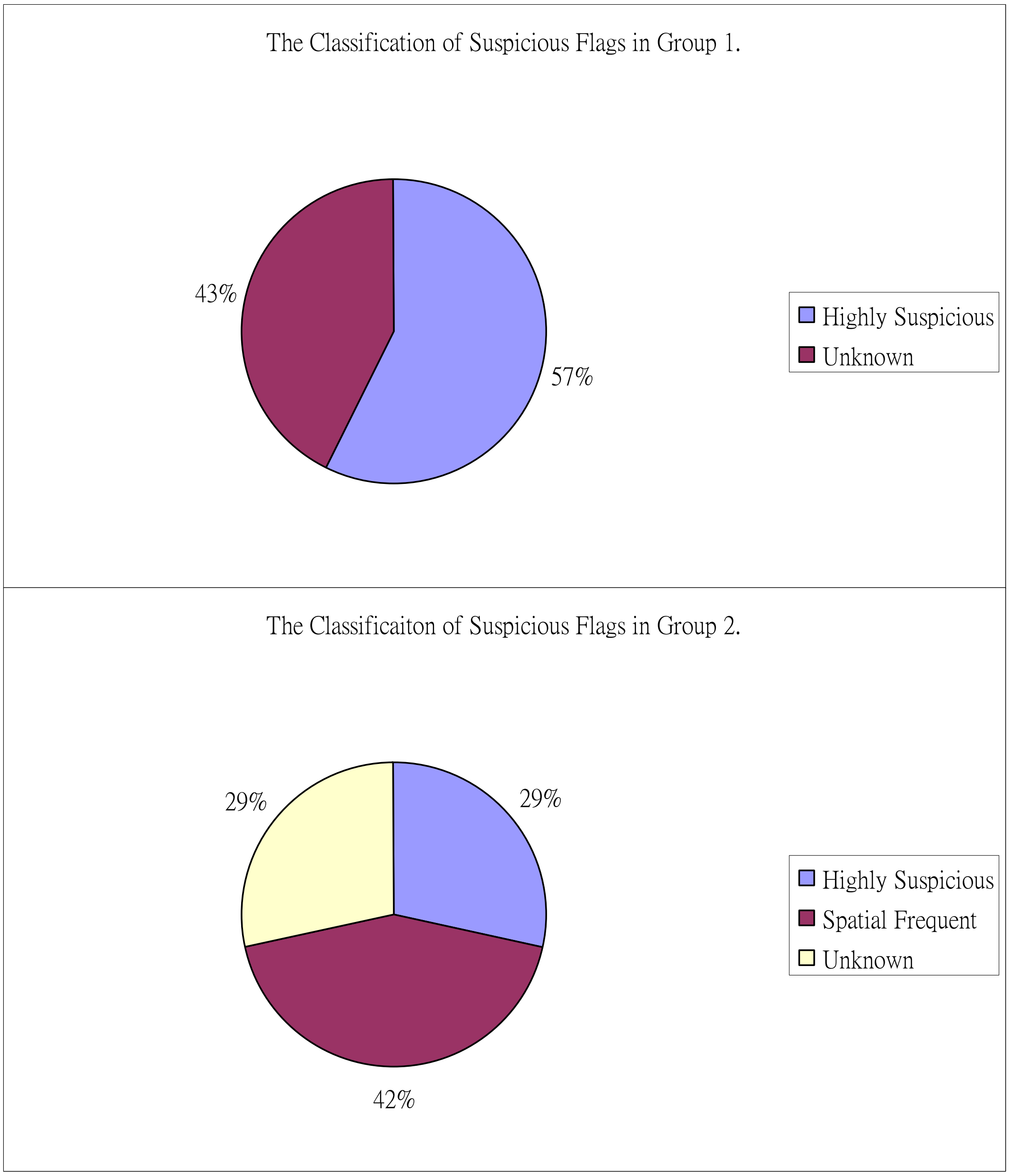


Figure 7.4: Exp2 - Observations of percentages of different suspicious flags in each sensor group

Chapter 8: Concluding Remarks

In this thesis, a **Knowledge-based framework for Collaborative Discovering of Suspicious Network Behaviors** is proposed to integrate most analytical algorithms which is used for alert transformation, alert correlation, alert aggregation and alert filtering. With knowledge-based approach, it is possible to assist administrators to select appropriate methods for different requirements of them and is easy to replace original analytical algorithms with new methods provided by other experts. Besides, integrated alert transactions can be analyzed on different concept levels of multiple dimensions in the data cube for discovering intrusion patterns with OLAP and data warehouse technique.

The proposed framework consists of three phases: **Data Preprocessing Phase, Alert Filtering Phase and Collaborative Analysis Phase**. In Data Processing Phase, a **Sensor Grouping (SG)** algorithm is proposed to assist administrators to divide sensors into groups with specific system and network profiles, and these groups are bases of target sensors in the Collaborative Analysis Phase; besides, an **Alert Format Transformation (AFT)** algorithm is proposed to assist administrators to transform IDS alerts of these groups into alert transactions with specific appropriate data formats according to requirements in the Collaborative Analysis Phase. Because of numerous of false alerts, an **Alert Filtering Method Selection (AFMS)** in Alert Filtering Phase is proposed to assist administrators to construct a appropriate **Filter Model (FM)** of sensors in specific group to filter most false alerts, and the results of this phase are seem as reliable data sources for the Collaborative Analysis Phase. In Collaborative Analysis Phase, an **Intra-Group Collaborative Analysis Selection (IGC-A)** algorithm is proposed to assist administrators to analyze each alert patterns with appropriate methods for specific attack types in one specific sensor group; finally, an **Inter-Group Collaborative Behavior Sharing (IGC-B)** algorithm is proposed to classify the results into aggregated information for administrators as references of intrusion defense in the viewpoint of specific sensor groups with similar backgrounds and behaviors.

To verify the feasibility of this knowledge-base framework, we propose corresponding algorithms in each phase for examples, and we use a data set as our data source to test the performance of these algorithms. As shown in Chapter 7, we can obtain useful information of suspicious alert patterns about novel intrusions.

There are two issues that we didn't discussed in this thesis will restrict the ability of our knowledge-based framework for CDSNB. First, the number of analysis algorithms in each phase would affect the results of this methodology. With the rapid and varied evolution of Internet intrusions, it is necessary to develop corresponding analysis methods to discover novel intrusion patterns effectively. OS system anomaly alarms are also important in complete intrusion lifecycles, and associating IDS alert information with these system alarms is meaningful for discovering suspicious system and network behaviors. Most intrusions not only cause suspicious network behaviors but also lead to some system anomalies such as executing a backdoor process. Some existing tools may be able to detect some unusual system states successfully, and include this system anomaly information in our knowledge-based framework will make this methodology more powerful and full-scale.

References

[1] Alharby, A. and Imai, H. (2005) "IDS False Alarm Reduction Using Continuous and Discontinuous Patterns", Proceedings of ACNS 2005, 2005, pp.192-205.

[2] Valdes, A. and Skinner, K. (2001) "Probabilistic Alert Correlation", Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection, 2001, pp.54-68.

[3] Morin, B. and Debar, H. (2003) "Correlation of Intrusion Symptoms: an Application of Chronicles", Proceedings of the 6th symposium on Recent Advances in Intrusion Detection (RAID 2003), September 2003.

[4] Cabrera, J. B. D., Lewis, L., Qin, X., Lee,W., Prasanth, R. K., Ravichandran, B. and Mehra, R. K. (2001) "Proactive detection of distributed denial of service attacks using MIB traffic variables - A feasibility study.", Proceedings of the 7th IFIP/IEEE International Symposium on Integrated Network Management, 2001.

[5] Erhard, W., Gutzmann, M. M. and Libati, H. M. (2000) "Network Traffic Analysis and Security Monitoring UniMon", Proceeding of the IEEE Conference on High Performance Switching and Routing, 2000, ATM 2000, pp 439-46.

[6] Cuppens, F. and Miege, A. (2002) "Alert correlation in a cooperative intrusion detection framework", Proceedings of the 2002 IEEE Symposium on Security and Privacy, May 2002.

[7] Goldman, R. P., Heimerdinger, W., Harp, S. A., Geib, C. W., Thomas, V. and Carter, R. L. (2001) "Information Modeling for Intrusion Report Aggregation", In DARPA Information Survivability Conference and Exposition II, 2001.

[8] Debar, H. and Wespi, A. (2001) "The intrusion-detection console correlation mechanism", In 4th International Symposium on Recent Advances in Intrusion Detection (RAID 2001), October 2001.

[9] Hsin, W. Y. (2005) "A Study of Alert-Based Collaborative Defense", National Chiao Tung University, Master Thesis, 2005.

[10] Chen, J., DeWitt, D. J., Tian, F. and Wang, Y. (2000) "NiagaraCQ: A scalable continuous query system for internet databases", Proceedings of ACM SIGMOD 2000, 2000, pp.379-390.

[11] Clement, L. Y. S. (2003) "Log Analysis as an OLAP Application - A Cube to Rule Them All", Practical assignment for GIAC GSEC certification, June 2003.

[12] Sabhnani, M. and Serpen, G. (2003) "Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection", Proceedings of the International Conference on Machine Learning; Models, Technologies and Applications. MLMTA'03, Jane 23-26, 2003.

[13] Shin, M. S., Kim, E. H. and Ryu, K. H. (2004) "False Alarm Classification Model for Network-Based Intrusion Detection System", Proceedings of IDEAL 2004, 2004, pp.259-265.

[14] Park, K. and Lee, H. (2001) "On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law Internets", Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, Aug. 2001.

[15] Porras, P. A., Fong, M. W. and Valdes, A. (2002) "A Mission-Impact-Based Approach to INFOSEC Alarm Correlation", Lecture Notes in Computer Science, Proceedings Recent Advances in Intrusion Detection, 2002, pp.95-114.

[16] Ning, P., Cui, Y. and Reeves, D. S. (2002) "Constructing attack scenarios through correlation of intrusion alerts", 9th ACM Conference on Computer and Communications Security, November 2002.

[17] Ning, P., Xu, D., Healey, C. G. and Amant, R. A. St. (2004) "Building attack scenarios through integration of complementary alert correlation methods", Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS'04), February 2004.

[18] Agrwal, R. and Srikant, R. (1995) "Mining Sequential Patterns", Proc. of the 11th Int'l Conference on Data Engineering, March 1995.

[19] Madden, S. R., Shah, M. A. and Hellerstein, J. M. (2002) "Continuously adaptive continuous queries over streams", Proceedings of ACM SIGMOD 2002, 2002.

[20] Cheung, S., Lindqvist, U. and Fong, M. W. (2003) "Modeling multistep cyber attacks for scenario recognition", Proceedings of the Third DARPA Information Survivability Conference and

Exposition (DISCEX III), April 2003.

[21] Srikant, R. and Agrawal, R. (1996) "Mining sequential patterns: Generalizations and performance improvements", Proc. of the Fifth Int'l Conference on Extending Database Technology (EDBT), 1996.

[22] Tseng, Y. C. (2004) "Monitoring Network Intrusion by OLAP and Data Mining", National Chiao Tung University, Master Thesis, 2004.

[23] Symantec Corp. (2006) "Symantec Internet Security Threat Report: Trends for July 05-Decamber 05" Volume IX, Published March 2006,

URL: http://www.symantec.com/index.htm.

[24] Basic Analysis and Security Engine (BASE),

URL: http://secureideas.sourceforge.net/, 2005.

[25] CERT Coordination Center, URL: http://www.cert.org/, 2006.

[26] DRAMA Expert System, CORETECH Inc.,

URL: http://www.coretech.com.tw/c_DRAMA.htm, 2006.

[27] Snort Intrusion Detection/Prevention System, URL: http://www.snort.org/, 2006.

[28] Taiwan Computer Emergency Response Team/Coordination Center (TWCERT/CC),

URL: http://www.cert.org.tw/, 2006.

Appendix

Snort Alert Transactions of alert warehouse for Experiment in order of timestamp (on 2005-12-14).

AlertID	Signature	Timestamp	Src_IP	Src_Port	Dst_IP	Dst_Port	Protocol
1	X	17:00:33	140.113.23.203	456	x.x.x.1	15104	ICMP
2	A	17:00:54	140.113.23.104	168	x.x.x.1	1269	TCP
3	X	17:01:03	140.118.26.54	178	x.x.x.2	267	ICMP
4	G	17:01:11	84.130.236.57	352	x.x.x.6	842	TCP
5	X	17:01:15	140.113.23.203	64765	x.x.x.8	13546	Raw IP
6	X	17:02:00	68.217.2.160	4046	x.x.x.3	695	ICMP
7	Z	17:03:15	148.233.0.157	203	x.x.x.3	445	TCP
8	H	17:03:26	203.67.195.180	4407	x.x.x.6	445	TCP
9	G	17:03:59	140.113.23.203	3550	x.x.x.7	445	TCP
10	Y	17:04:23	83.27.74.46	1515	x.x.x.2	445	Raw IP
11	B	17:04:31	140.118.26.54	3654	x.x.x.1	705	TCP
12	X	17:05:10	84.130.236.57	16538	x.x.x.4	161	TCP
13	Y	17:05:29	140.113.23.203	3538	x.x.x.1	163	TCP
14	X	17:06:15	68.217.2.160	486	x.x.x.5	468	TCP
15	G	17:06:20	148.233.0.157	6897	x.x.x.5	789	TCP
16	X	17:06:55	82.134.206.104	266	x.x.x.8	665	Raw IP
17	I	17:07:58	140.113.69.180	6884	x.x.x.6	445	TCP
18	Y	17:09:13	140.113.23.104	7963	x.x.x.4	665	TCP
19	Y	17:09:26	140.118.26.54	769	x.x.x.5	789	TCP
20	B	17:10:15	84.130.236.57	936	x.x.x.2	445	ICMP
21	X	17:10:36	201.144.78.3	267	x.x.x.1	445	TCP
22	Z	17:10:55	220.159.55.115	64765	x.x.x.2	705	Raw IP
23	S	17:12:22	40.121.222.183	4046	x.x.x.8	161	ICMP
24	A	17:13:15	68.217.2.160	203	x.x.x.3	289	TCP
25	X	17:14:59	148.233.0.157	4407	x.x.x.4	639	TCP
26	H	17:16:01	203.67.195.180	3550	x.x.x.7	367	TCP
27	X	17:16:45	140.113.23.203	456	x.x.x.2	4544	Raw IP
28	C	17:17:22	83.27.74.46	168	x.x.x.1	705	TCP
29	H	17:17:56	140.118.26.54	178	x.x.x.5	445	Raw IP
30	Y	17:19:12	84.130.236.57	3460	x.x.x.4	445	ICMP
31	X	17:26:20	24.208.151.22	6872	x.x.x.5	445	TCP
32	I	17:30:26	201.19.135.138	699	x.x.x.7	267	TCP

33	X	17:32:22	84.130.236.57	3654	x.x.x.3	445	TCP
34	Y	17:33:33	201.144.78.3	16538	x.x.x.2	445	TCP
35	Y	17:33:33	220.159.55.115	3538	x.x.x.3	705	Raw IP
36	X	17:40:12	40.121.222.183	486	x.x.x.7	161	TCP
37	Y	17:40:59	148.233.0.157	6897	x.x.x.7	769	Raw IP
38	Y	17:50:10	203.67.195.180	64765	x.x.x.1	936	ICMP
39	Z	17:51:20	140.113.23.203	4046	x.x.x.3	267	ICMP
40	I	17:52:36	140.113.23.104	1364	x.x.x.5	645	TCP
41	Y	17:53:50	140.118.26.54	63877	x.x.x.2	446	ICMP
42	X	17:54:40	84.130.236.57	6885	x.x.x.7	601	TCP
43	Z	17:55:36	140.113.23.203	4566	x.x.x.2	445	Raw IP
44	C	17:55:55	68.217.2.160	203	x.x.x.1	445	ICMP
45	Y	17:56:10	148.233.0.157	4407	x.x.x.7	667	TCP
46	Y	17:57:38	148.233.0.157	3550	x.x.x.5	352	TCP
47	C	17:59:03	82.134.206.104	348	x.x.x.2	445	ICMP
48	X	18:00:33	140.113.69.180	3453	x.x.x.3	225	ICMP
49	X	18:01:20	71.246.54.246	2687	x.x.x.6	80	TCP
50	J	18:03:11	84.167.211.153	7963	x.x.x.5	665	TCP
51	S	18:04:26	61.31.174.199	769	x.x.x.8	789	TCP
52	X	18:04:57	140.113.160.128	936	x.x.x.2	445	Raw IP
53	X	18:05:26	70.121.251.192	267	x.x.x.1	80	TCP
54	X	18:06:17	84.167.211.153	2364	x.x.x.7	80	TCP
55	Y	18:10:30	84.167.211.153	1687	x.x.x.3	445	TCP
56	V	18:11:44	83.27.74.46	4557	x.x.x.8	196	TCP
57	B	18:12:18	140.118.26.54	3538	x.x.x.7	80	TCP
58	D	18:14:30	84.130.236.57	486	x.x.x.2	769	Raw IP
59	Y	18:15:59	140.113.23.203	6897	x.x.x.7	936	TCP
60	Y	18:16:02	68.217.2.160	266	x.x.x.1	80	TCP
61	X	18:17:36	148.233.0.157	6884	x.x.x.4	445	TCP
62	Y	18:19:19	82.134.206.104	1151	x.x.x.6	80	ICMP
63	B	18:20:34	71.240.108.83	4225	x.x.x.3	268	TCP
64	K	18:21:32	68.217.2.160	4407	x.x.x.5	80	Raw IP
65	X	18:22:56	148.233.0.157	4046	x.x.x.7	445	TCP
66	E	18:24:21	203.67.195.180	64765	x.x.x.2	297	TCP
67	X	18:27:37	71.103.79.52	4046	x.x.x.6	445	TCP
68	Y	18:28:33	140.118.26.54	203	x.x.x.4	445	Raw IP
69	Y	18:29:14	84.130.236.57	4407	x.x.x.2	8080	TCP

70	Y	18:30:06	140.113.144.39	3550	x.x.x.6	936	TCP
71	X	18:31:33	70.111.98.123	6897	x.x.x.3	80	TCP
72	C	18:32:44	24.33.151.138	64765	x.x.x.7	80	ICMP
73	X	18:33:46	84.167.211.153	4046	x.x.x.1	161	TCP
74	T	18:34:51	24.174.30.166	16538	x.x.x.8	769	Raw IP
75	X	18:38:00	71.246.54.246	3538	x.x.x.6	80	ICMP
76	Y	18:39:07	84.167.211.153	486	x.x.x.1	445	TCP
77	Y	18:40:26	84.167.211.153	3447	x.x.x.7	445	ICMP
78	C	18:41:06	83.27.74.46	25558	x.x.x.3	80	TCP
79	F	18:43:55	140.118.26.54	6587	x.x.x.2	445	Raw IP
80	B	18:44:22	84.130.236.57	1364	x.x.x.4	705	ICMP
81	Y	18:46:24	140.113.23.203	9956	x.x.x.3	161	TCP
82	C	18:50:07	84.167.211.153	4046	x.x.x.8	445	TCP
83	X	18:51:05	71.246.54.246	203	x.x.x.2	267	TCP
84	X	18:56:23	68.217.2.160	352	x.x.x.4	80	Raw IP
85	Y	18:58:33	148.233.0.157	64765	x.x.x.2	289	TCP
86	Z	18:59:01	82.134.206.104	2687	x.x.x.1	639	TCP
87	Y	18:59:33	71.240.108.83	2387	x.x.x.6	367	TCP
88	Y	18:59:58	68.217.2.160	4407	x.x.x.4	445	TCP
89	X	19:00:03	60.40.12.206	4069	x.x.x.5	80	Raw IP
90	S	19:02:11	71.106.91.74	3360	x.x.x.8	665	ICMP
91	X	19:03:33	140.113.69.180	456	x.x.x.2	789	TCP
92	X	19:05:12	216.129.198.148	168	x.x.x.1	445	TCP
93	J	19:07:59	220.159.63.5	178	x.x.x.6	80	ICMP
94	Y	19:09:33	148.244.106.226	3460	x.x.x.3	80	TCP
95	I	19:10:53	71.106.91.74	2017	x.x.x.7	445	ICMP
96	D	19:11:22	12.227.172.223	16538	x.x.x.1	196	TCP
97	B	19:12:29	84.130.236.57	3538	x.x.x.5	705	Raw IP
98	X	19:13:45	140.113.23.203	486	x.x.x.8	161	ICMP
99	Y	19:15:37	68.217.2.160	6897	x.x.x.2	163	TCP
100	X	19:16:34	148.233.0.157	266	x.x.x.6	468	TCP
101	X	19:19:27	48.233.0.157	4932	x.x.x.4	789	ICMP
102	Y	19:22:26	71.241.166.36	3360	x.x.x.1	665	ICMP
103	K	19:26:29	216.129.198.148	352	x.x.x.6	445	TCP
104	Y	19:26:58	71.240.108.83	936	x.x.x.8	80	TCP
105	Z	19:27:46	58.157.110.9	64765	x.x.x.3	665	TCP
106	Z	19:28:04	60.40.12.206	3360	x.x.x.5	789	TCP

107	Z	19:28:55	4.153.50.101	64765	x.x.x.2	445	Raw IP
108	Y	19:31:37	157.157.242.20	4046	x.x.x.3	163	TCP
109	Y	19:32:06	140.113.23.203	2017	x.x.x.6	468	TCP
110	C	19:33:49	83.27.74.46	3360	x.x.x.5	789	TCP
111	X	19:38:41	140.118.26.54	1611	x.x.x.7	267	Raw IP
112	X	19:39:07	216.129.198.148	168	x.x.x.1	80	TCP
113	L	19:43:39	216.129.198.148	178	x.x.x.6	445	ICMP
114	Z	19:45:29	140.113.141.164	3360	x.x.x.3	705	TCP
115	Z	19:46:37	148.244.106.226	1326	x.x.x.4	445	ICMP
116	X	19:48:41	82.134.206.104	4046	x.x.x.2	705	TCP
117	Z	19:51:29	140.113.69.180	4069	x.x.x.8	161	TCP
118	Y	19:52:36	140.113.23.104	3666	x.x.x.1	175	TCP
119	X	19:53:19	140.118.26.54	4932	x.x.x.5	288	Raw IP
120	F	19:53:29	84.130.236.57	6884	x.x.x.3	469	TCP
121	X	19:54:44	162.135.16.6	7963	x.x.x.4	665	TCP
122	X	19:55:21	216.129.198.148	769	x.x.x.6	445	TCP
123	Y	19:56:39	148.233.0.157	936	x.x.x.2	665	TCP
124	Y	19:57:37	62.6.163.135	267	x.x.x.6	789	TCP
125	Z	19:57:50	216.129.198.148	3360	x.x.x.4	445	Raw IP
126	E	19:58:03	66.37.74.183	2017	x.x.x.1	445	TCP
127	Z	19:59:29	71.106.91.74	1326	x.x.x.5	161	TCP
128	X	20:00:55	151.201.7.251	4809	x.x.x.4	80	TCP
129	Z	20:02:37	69.236.199.59	63875	x.x.x.2	138	TCP
130	Y	20:03:51	69.19.228.43	3008	x.x.x.1	445	TCP
131	X	20:04:49	66.65.204.185	3579	x.x.x.3	445	ICMP
132	X	20:05:22	140.118.26.54	4340	x.x.x.6	80	TCP
133	J	20:08:11	84.130.236.57	3008	x.x.x.7	445	TCP
134	D	20:09:03	221.169.91.177	63875	x.x.x.3	705	TCP
135	X	20:11:55	4.252.246.167	3237	x.x.x.8	161	Raw IP
136	A	20:13:26	203.67.195.180	1049	x.x.x.4	445	TCP
137	Z	20:14:33	140.113.23.203	3888	x.x.x.1	267	ICMP
138	X	20:15:06	69.236.199.59	4617	x.x.x.8	80	TCP
139	X	20:16:24	140.118.26.54	1945	x.x.x.2	524	ICMP
140	Z	20:17:26	84.130.236.57	3446	x.x.x.6	445	Raw IP
141	Z	20:19:01	4.252.246.167	4810	x.x.x.4	445	TCP
142	E	20:19:55	71.246.54.246	62603	x.x.x.3	705	ICMP
143	L	20:23:09	84.167.211.153	61621	x.x.x.5	161	TCP

144	Z	20:25:22	151.201.7.251	3579	x.x.x.3	769	Raw IP
145	B	20:26:19	140.113.127.10	4046	x.x.x.6	936	ICMP
146	K	20:27:39	66.65.204.185	203	x.x.x.7	267	TCP
147	Y	20:29:56	216.129.198.148	4407	x.x.x.1	645	TCP
148	X	20:33:19	71.99.160.206	486	x.x.x.6	445	ICMP
149	Y	20:35:02	69.236.199.59	6897	x.x.x.2	80	ICMP
150	B	20:38:26	140.118.26.54	266	x.x.x.4	445	TCP
151	C	20:41:23	84.130.236.57	4810	x.x.x.6	445	TCP
152	Z	20:42:10	140.113.23.203	2950	x.x.x.1	667	TCP
153	X	20:43:29	68.217.2.160	3538	x.x.x.3	352	ICMP
154	T	20:45:18	4.252.246.167	486	x.x.x.8	445	TCP
155	Z	20:46:53	71.99.160.206	1687	x.x.x.2	225	TCP
156	F	20:47:29	66.65.204.185	4557	x.x.x.3	80	TCP
157	X	20:48:24	221.169.91.177	3538	x.x.x.4	445	TCP
158	F	20:50:06	84.130.236.57	7963	x.x.x.1	639	Raw IP
159	Z	20:51:17	140.113.23.203	769	x.x.x.6	367	TCP
160	C	20:52:31	69.236.199.59	936	x.x.x.4	4544	TCP
161	Z	20:54:41	203.67.195.180	6897	x.x.x.3	705	TCP
162	L	20:56:52	140.113.23.203	266	x.x.x.7	445	ICMP
163	Y	20:57:55	71.99.160.206	6884	x.x.x.2	445	TCP
164	Z	20:58:31	159.101.47.33	16872	x.x.x.4	445	TCP