View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by DSpace@IZTECH Institutional Repository

# Development of a Quality Assurance Prototype for Intrusion Detection Systems

**By**
**Ulaş YÜKSEL**

**A Dissertation Submitted to the Graduate School in Partial Fulfillment of the Requirements for the Degree of**

**MASTER OF SCIENCE**

**Department:** Computer Engineering
**Major:** Computer Software

**İzmir Institute of Technology**
**İzmir, Turkey**

**June, 2002**

We approve the thesis of **Ulaş YÜKSEL**

**Date of Signature**

…… **28.06.2002**

**Asst. Prof. Tuğkan TUĞLULAR, Ph.D**.
Supervisor
Department of Computer Engineering

…… **28.06.2002**

**Assoc. Prof. Ahmet KOLTUKSUZ, Ph.D.**
Department of Computer Engineering

…… **28.06.2002**

**Assoc. Prof. Oğuz DİKENELLİ, Ph.D.**
Ege University
Department of Computer Engineering

…… **28.06.2002**

**Prof. Sıtkı AYTAÇ, Ph.D.**
Department of Computer Engineering
Head of Department

# ACKNOWLEDGEMENTS

First and foremost, I would like to sincerely thank my advisor, Asst. Prof. Tugkan TUGLULAR, Ph.D. for directing me to the right way and sharing his knowledge. I would like to thank Assoc. Prof. Ahmet KOLTUKSUZ, Ph.D. for getting me interested in Computer Security and giving me the bases of security point of view. I would also like to thank all the people at İzmir Institute of Technology, Department of Computer Engineering, for always being friendly, supportive and helpful.

Finally, I thank my family, whose constant love and support give me confidence.

# ABSTRACT

Quality assurance is an essential activity for any business interacting with consumers. There are considerable number of projects going on to develop intrusion detection systems (IDSs). However, efforts to establish standards and practices to ensure the quality of such systems are comparatively less significant. The quality assurance activities for IDSs should ensure the conformance of explicitly stated functional and performance requirements as well as implicit characteristics that are expected from information security tools.

This dissertation establishes guidelines to review, evaluate and possibly to develop an IDS. To establish guidelines, generic IDS and software requirements, software quality factors and design principles are used which are available in related literature and these requirements are presented both on developed generic IDS model and in Common Criteria Protection Profile format. First, the guidelines are developed, then they are implemented on a specific IDS product evaluation.

# ÖZ

Müşteri ile iletişim halinde bulunulan ve müşterilere ürün sunulan her iş alanında, kalite güvence çalışmaları önem kazanmaktadır. Değişik işletim sistemleri için kayda değer sayıda Nüfuz Tespit Sistem'i (NTS) geliştirme çalışması bulunmasına karşın, bu sistemlerin kalitesini ortaya koyacak olan standartların geliştirilmesine, aynı derecede önem verilmemektedir. NTSleri için kalite güvence faaliyetleri, ürüne özel fonksiyonel ve performans gereksinimlerinin yerine getirildiğini ve aynı zamanda genel güvenlik araçları için geçerli gereksinimlerin sağlandığını göstermelidir.

Bu tez çalışmasında, bir NTSnin geliştirilmesi ve değerlendirilmesi için kullanılabilecek kılavuzlar ortaya konulmuştur. Bu kılavuzların geliştirilmesi için, genel NTSleri ve yazılım gereksinimleri, yazılım kalitesi ve geliştirme prensipleri hakkındaki mevcut literatür kullanılmıştır. Ortaya konulan gereksinimler, hem geliştirilen genel NTS modeli üzerinde, hem de Genel Kriterler (Common Criteria) Koruma Profili (Protection Profile) biçiminde sunulmuştur. Son olarak geliştirilen kılavuzlar ve gereksinimler, bir NTS ürününün değerlendirilmesi için kullanılmıştır.

# TABLE OF CONTENTS

LIST OF FIGURES ..... viii
LIST OF TABLES ..... ix
1 INTRODUCTION ..... 1
2 OVERVIEW AND METHODOLOGY ..... 3
2.1 Computer Intrusions ..... 3
2.2 Intrusion Detection Systems ..... 3
2.3 Client Server Model ..... 6
2.4 Security Software Quality Assurance ..... 8
2.4.1 Parts of the Common Criteria ..... 10
2.4.2 Protection Profile and Common Criteria Usage ..... 13
2.5 Methodology ..... 14
3 GENERIC INTRUSION DETECTION SYSTEM MODEL AND REQUIREMENTS ..... 19
3.1 Graphical User Interface ..... 21
3.2 Management Server ..... 21
3.3 Database Server ..... 23
3.4 Intrusion Detection Module ..... 24
3.4.1 Collector ..... 24
3.4.2 Analyzer ..... 25
3.5 System Requirements ..... 26
3.5.1 Development ..... 26
3.5.2 Installation, Deployment and Update Capability ..... 27
3.5.3 Configuration ..... 28
3.5.4 Performance ..... 29
3.5.5 Testing ..... 30
3.5.6 Documentation ..... 32
3.5.7 Security ..... 33
4 INTRUSION DETECTION SYSTEM PROTECTION PROFILE FOR PC LABORATORY LAN ENVIRONMENT ..... 34
4.1 Protection Profile Introduction ..... 34
4.1.1 Identification ..... 34
4.1.2 Overview ..... 34
4.2 Target of Evaluation Description ..... 34
4.2.1 Architecture and Working Principles ..... 34
4.2.2 Scope and Boundaries of the Target of Evaluation ..... 34
4.3 Target of Evaluation Security Environment ..... 35
4.3.1 Assumptions ..... 38
4.3.2 Threats ..... 39
4.4 Security Objectives ..... 44
4.4.1 Information Technology Security Objectives ..... 44
4.4.2 Security Objectives for the Environment ..... 45
4.5 Information Technology Security Requirements ..... 46
4.5.1 Target of Evaluation Security Requirements ..... 47
4.5.2 Security Requirements for the Information Technology Environment ..... 58
4.6 Rationale ..... 58
4.6.1 Rationale For Information Technology Security Objectives ..... 58
4.6.2 Rationale For Security Objectives for the Environment ..... 59
4.6.3 Rationale For Security Requirements ..... 60
5 INTRUSION DETECTION SYSTEM PROTECTION PROFILE IMPLEMENTATION ON PRODUCT EVALUATION ..... 62
5.1 RealSecure Version 6.0 ..... 62
5.2 Evaluation of RealSecure Version 6.0 ..... 63
6 CONCLUSION AND DISCUSSION ..... 72
REFERENCES ..... 75
APPENDIX ..... 80
APPENDIX A IDS ASSESSMENT TABLE ..... 80
APPENDIX B COVERAGE OF SOFTWARE ENGINEERING STANDARDS ..... 82
APPENDIX C COMMON CRITERIA SECURITY ASSURANCE REQUIREMENTS ..... 83

APPENDIX D PROPOSED COMMON CRITERIA INTRUSION DETECTION CLASS ..... 93

# LIST OF FIGURES

Figure 2.1: Example Three Tier Client-server Architecture. ....7
Figure 2.2: Software Acquisition Life Cycle. ....9
Figure 2.3: IDS Requirement and PP Development Methodology. ....17
Figure 3.1: Intrusion Detection System Model. ....19
Figure 4.1: PcLab Topology. ....36
Figure 4.2: A Summary of Possible Attack Description. ....40
Figure 4.3: TOE Threats. ....41
Figure 4.4: IT System Threats. ....42
Figure 5.1: Evaluation Process and Results. ....64
Figure D.1: Intrusion detection class decomposition. ....93

# LIST OF TABLES

Table 3.1: Asset Database Table. ..... 23
Table 3.2: Events Table. ..... 25
Table 3.3: Intrusion Database Table. ..... 26
Table 4.1: PcLab Specification Summary ..... 36
Table 4.2: TOE Threats. ..... 42
Table 4.3: IT System Threats. ..... 43
Table 4.4: Threat to be Countered by Operating Environment. ..... 43
Table 4.5: Auditable Events. ..... 49
Table 4.6: Restated TOE Security Assurance Requirements ..... 57
Table 4.7: Summary of Mappings Between Threats, Policies and IT Security Objectives. ..... 59
Table 4.8: Summary of Mapping Between Threats and Security Objectives for the Environment. ..... 59
Table 4.9: Summary of Mappings Between Functional Requirements and Objectives for the TOE. ..... 60
Table 5.1: ISS RealSecure Functional Requirements Assessment Table. ..... 65
Table 5.2: ISS RealSecure Security Assurance Requirements Assessment Table. ..... 68
Table A.1: IDS Assessment Table. ..... 81
Table B.1: Software Engineering Standards Table. ..... 82

# Chapter 1

# INTRODUCTION

Before the “internet age”, networks were simpler and there were fewer threats. A single knowledgeable network manager could check the security logs and look for anomalies in those days. As networks grew, customers moved to automated solutions. First came automated log checkers. Then agents are used to watch for suspicious behavior. Every time a new attack was spotted the computer security industry responded with new attack signatures. Networks are becoming complex, larger, more distributed and have more access points, which result in more vulnerabilities. There is also more at risk as companies rush to make valuable corporate assets available over networks and business systems. Heavy reliance on computers and increased network connectivity increased the risk of potential damage from attacks that can be launched from remote locations. Not only the networks but also operating systems and applications are also growing, becoming more “powerful and complex”. This complexity makes these systems more likely to contain bugs, some of which can be exploitable by the attackers to gain access to system and its data (Webster, 1998). Current security measures such as firewalls, security policies, and encryption are not sufficient to prevent the compromise of private computers and networks that a fundamentally new approach is called for. (Das, 2000; Internet Security Systems, 1999).

Then, intrusion detection systems have become an essential component of computer security to supplement existing defences. This field underwent explosive growth in the last couple of years. Even after the several early releases, there still remain at least seventeen extant products that claim to provide intrusion detection in a networked environment (Internet Security Systems, 1999). However, standards for such IDSs have not been established yet, though there are some on going projects. Thus, this thesis aims to bring up high level standards for a generic IDS model monitoring a computer system to detect and react to intrusions. A computer system may range from a single computer to a network. An IDS performs following functions: collecting the information regarding computer system activity, data [and vulnerabilities]; analysing and reporting of the collected information, finally reacting detected intrusions.

This thesis can be used basically by three groups of people: IDS consumers, developers and evaluators. In order to see the requirements of an IDS and to select the best-suited product to their organizational needs, consumers can refer to this prototype. Consumers may prepare a simple checklist or use the one given in Appendix as well to see the capabilities of available IDS products.

Developers can use this thesis to write the specifications of their IDS product before the production phase. During and after the production phase they can apply to this work to evaluate their product. Finally the thesis can be used by evaluators when forming judgement about an IDS product. Note that this work does not give the procedures to be followed in either evaluation or development phases.

This thesis is organized in five chapters. Chapter 2 provides background information about intrusion, intrusion detection, software quality assurance, client-server architecture and the methodology followed in this thesis. Chapter 3 classifies and documents the IDS requirements on developed generic IDS Model. Chapter 4 represents IDS requirements in Common Criteria Protection Profile (PP) format. For this purpose a draft IDS Protection Profile is written for sample LAN environment. This IDS PP is implemented on a commercially available IDS called ISS RealSecure in Chapter 5. Finally, in Chapter 6 conclusions and suggestions are provided for future work on quality assurance and standardization of IDS.

# Chapter 2

# OVERVIEW AND METHODOLOGY

## 2.1 Computer Intrusions

Many researchers define intrusion as "any set of action that attempts to compromise the integrity, confidentiality or availability of a resource" (Heady et al. 1990 as cited in Kumar, 1995; Lodin, 1999). Anderson (1980 as cited in Kumar, 1995) uses “threat” with the same meaning and defines it as the potential possibility of deliberate unauthorized attempt to access or manipulate information or to render a system unreliable or unusable. In this thesis, the first definition is accepted. Based on that approach, a computer intrusion is re-defined as any action attempting to break into or misuse the computer system which may lead to loss of the integrity, confidentiality or availability of a computer system resource.

A computer attack can be defined as the any malicious activity against the computer system. Attacks can be trojan horses, viruses, actions aiming the denial of any computer system service and even physical damages. According to above definitions, intrusions are a subset of attacks. However, since this thesis focuses on the computer intrusions, the terms "computer intrusion", "intrusion", also "computer attack" and "attack" are used interchangeably to mean “intrusion” as it is defined above.

There are two types of potential intruders, outsiders and insiders (or legitimate users). Outsiders are the most publicized form of intruders and mostly called as crackers or hackers. They attack the computer system via networks. Inside intruders are authorized users of the organization. Those people has potential for causing the greatest damage on computer system because they have physical access, some level of genuine privilege, knowledge about the location of the valuable data, the computer system and its security mechanisms.

## 2.2 Intrusion Detection Systems

As their names imply, Intrusion Detection Systems (IDSs) are the systems that attempt to detect intrusions and/or defend against them. IDSs react to the potential attacks automatically. While these reactions can be warnings to the system security

officer; they can also be recovery actions, defensive actions (e.g., shutting down the attacked system service) or any combination of them. IDSs are based on the hypothesis that monitoring and analyzing network transmissions, system audit records, application audit records, system configuration, data files, and other information can detect intrusions.

Generally speaking, current IDSs can be classified as either anomaly detectors or misuse detectors (or both). Anomaly detectors are based on the observation of deviations from normal usage/behavior. IDS stores the individual or group of users'/subjects' normal behavior profiles and flags the actions deviating from these profiles. These profiles are constructed from historical data collected over a period of normal operation. By this method IDSs can catch the masquerading attacks where an intruder logs into the computer system by someone else's password or the legitimate users misusing their privileges. The disadvantage of this method is that it does not work if the user's activity is not stable and/or is overlapping with attacker activity. If the user's profiles is changing very fast and his/her actions are very similar to the an attacker's (like system administrator's actions) then the anomaly detection system may not differ a legitimate user and attacker or normal user activity and legitimate user's misusing activity.

Anomaly detection includes following two basic measures and techniques. The first one is threshold detection, in which certain attributes of user and system behavior are expressed in terms of counts, with some level established as permissible. Such behavior attributes can include but not limited to, the number of files accessed by a user in a given period of time, the number of failed attempts to login to the system, the amount of CPU utilized by a process.

The second one is statistical measures, both parametric, where the distribution of the profiled attributes is assumed to fit a particular pattern, and non-parametric, where the distribution of the profiled attributes is “learned” from a set of historical values, observed over time. Other measures, including neural networks, genetic algorithms, and immune system models are also available but they are not used in current IDSs in general.

IDSs with misuse detectors "watches for the indications of specific, precisely representable techniques of computer system abuse" (Puketza et al. 1994). IDS watches

the computer system and searches occurrence of these "signatures" that are stored as known intrusion techniques. As only known intrusions are detected, IDS must be constantly updated with signatures. Misuse detectors are effective at detecting attacks without generating an overwhelming number of false alarms. They can quickly and reliably diagnose the use of a specific attack tool or technique. This can help security managers prioritize corrective measures.

These above approaches are major but there are some other techniques for detection. Integrity checking is one of them. In this method IDS checks the integrity and changes of the prior system files and configurations. If the critical and important system files has been changed, IDS reacts.

For all above intrusion detection purposes most traditional IDSs have taken either a network-based or a host-based approach. Network-based IDSs evaluate information captured from network traffic. They analyze packets travelling across the network. Network-based IDS comprise software that is installed on dedicated workstations that are placed at critical junctions of a network (e.g., just outside a firewall, in front of a Web server or in front of an e-mail server). They "sniff" (capture and read) the stream of IP packets travelling across the network. A few well-placed network-based IDSs can monitor a large network. Deployment of network-based IDSs has little impact upon an existing network but they may have difficulty processing all packets in a large or busy network. Therefore, they may fail to recognize an attack launched during periods of high traffic. Network-based IDSs are usually passive devices that listen on a network wire without interfering with the normal operation of a network. Thus, it is usually easy to retrofit a network to include network-based IDSs with minimal effort (Schepers, 1998; Barber, 2001).

Host-based IDSs evaluate information found on some kind of host computer. This may include contents of operating system, file system, applications, and perhaps, other similar information. Host computers may include user workstations (including specialized applications such as web browsers), peripherals (such as printers), specialized servers such as web servers, or network components (such as firewalls, routers, and switches). Host-based IDSs use software modules that are installed on each monitored host. They access and read logs and audit records of interest (Lodin, 1999).

Host-based IDSs, with their ability to monitor events local to a host, may detect attacks that cannot be seen by a network-based IDS and they can often operate in an environment in which network traffic is encrypted, when the host-based information sources are generated before data is encrypted and/or after the data is decrypted at the destination host. Host-based IDSs are comparatively harder to manage, as information must be configured and managed for every host monitored and they can be disabled by certain denial-of-service attacks. When host-based IDSs use operating system audit trails as an information source, the amount of information can be immense, requiring additional local storage on the system. Additionally, they use the computing resources of the hosts they are monitoring, therefore inflicting a performance cost on the monitored systems (Durst, et al. 1999).

Each approach, network and host-based, has its strengths and weaknesses, but they are both complementary, and a truly effective IDS will employ both technologies (Schepers, 1998).

### 2.3 Client Server Model

A large system or software can be decomposed into sub-systems and modules according to service or function provided by them. “The initial design process of identifying these sub-systems and establishing a framework for sub-system control and communication is called architectural design” (Sommerville, 1995, p. 227). Sommerville (1995) also states that architectural design usually comes before detailed system specification and includes following activities:

- System structuring: The system is structured into a number of principal sub-systems where a sub-system is an independent software unit. Communications between sub-systems are identified.
- Control modelling: A general model of the control relationship between the parts of the system is established.
- Modular decomposition: Each identified sub-system is decomposed into modules. The types and interconnections between modules are defined.

Architectural design is based on a particular model in general. In this thesis client-server model is used for system structuring. Typically, this allows a user program at a host to act as a client issuing requests to programs and databases that may be

located at many remote-computing sites. The major components of client-server model are (Sommerville, 1995, p. 230):

- Set of stand-alone servers which provide specific services such as data and management.
- Set of clients, which call on these services.
- Infrastructure/network which allows communication between clients and servers.

There are some specific implementations of client-server model. One of those implementations is three-tier architecture, which is partially implemented in the Chapter 3 on generic IDS model. In a three-tier architecture, there are mainly three tiers; the user interface (client), data management (server) components, and the third tier (middle tier server) between the previous two tiers. This middle tier provides basically process management by providing functions such as queuing, application execution, and database staging.

A three-tier client/server architecture (Figure 2.1) includes user interfaces at top tier where user services (such as session, text input, dialog, and display management) reside. The third tier provides database management functionality and is dedicated to data and file services. The middle tier provides process management services (such as process development, process enactment, process monitoring, and process resourcing) that are shared by multiple applications. The middle tier server (also referred to as the application server) improves performance, flexibility, maintainability, reusability, and

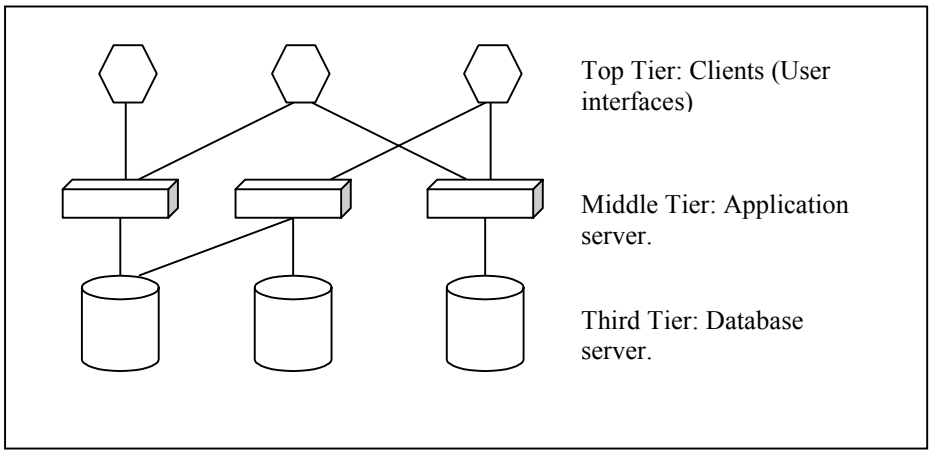


Figure 2.1: Example Three Tier Client-server Architecture.

scalability by centralizing process logic. Centralized process logic makes administration and change management easier by localizing system functionality so that changes must only be written once and placed on the middle tier server to be available throughout the systems (The Software Engineering Institute, 2002).

The most important advantage of client-server model is straightforward distribution that may be also an advantage for IDSs because IDSs collect, analyze and process some computer system and network data at different distributed locations and hosts on the network. Thus, IDS model in Chapter 3 is based on client-server architecture and some more details about this model will be given in that chapter.

## 2.4 Security Software Quality Assurance

While assurance is defined as grounds for confidence that a deliverable meets its objectives, Software Quality Assurance (SQA) is defined as "a planned and systematic approach to the evaluation of the quality of and adherence to software product standards, processes, and procedures" (NASA-GB-A201, 1992). Additionally, in IEEE Std 730-1998, quality assurance is defined as "a planned and systematic pattern of all actions necessary to provide adequate confidence that the item or product conforms to established technical requirements" (IEEE Std-730, 1998). SQA includes the process of assuring that standards and procedures are established and are followed throughout the software acquisition life cycle which involves three activities: management, engineering and assurance which are illustrated in the Figure 2.2. Compliance with agreed-upon standards and procedures is evaluated through process monitoring, product evaluation, and audits where standards are the established criteria to which the software products are compared and procedures are the established criteria to which the development and control processes of a software are compared.

According to NASA Software Assurance Standard, the SQA process shall ensure that (NASA-STD-2201-93, 1992):

- Standards and procedures for management, engineering, and assurance activities are specified.
- Management, engineering, and assurance adhere to specified standards and procedures.
- All documentation and report formats and content descriptions are defined.

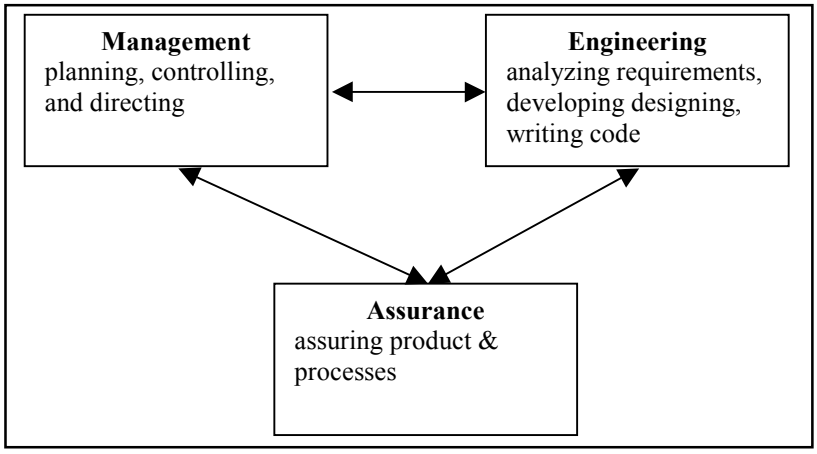


Figure 2.2: Software Acquisition Life Cycle.

- All plans (configuration management, risk management, assurance plan, management plan, etc.) are completed and implemented according to specified standards and procedures.
- The configuration management process functions according to approved procedures and that all baselined items are maintained under Configuration Management.
- Baseline control, configuration identification, configuration control, configuration status accounting, and configuration authentication activities are carried out.

NASA Software Assurance Standard also states that the SQA process shall include the following activities (NASA-STD-2201-93, 1992):

- Evaluation of specified standards and procedures.
- Audits of all management, engineering, and assurance processes, for example, Configuration Management.
- Reviews of all project documentation including reports, schedules, and records.
- Monitoring of formal inspections and formal reviews.
- Monitoring/witnessing of formal and acceptance-level software testing.

These and any other quality assurance activities are performed to provide adequate confidence that software product conforms to established technical requirements. For this confidence, the requirements of an individual software should have been developed first essentially.

There are series of efforts to develop criteria and set of requirements for specifying and evaluating IT security requirements. In the early 1980s the Trusted Computer System Evaluation Criteria (TCSEC) was developed in the United States. In

the succeeding decade, various countries began initiatives to develop evaluation criteria that built upon the concepts of the TCSEC but were more flexible and adaptable to the evolving nature of IT in general (Common Criteria Org.,2002b).

In Europe, the Information Technology Security Evaluation Criteria (ITSEC) version 1.2 was published in 1991 by the European Commission after joint development by the nations of France, Germany, the Netherlands, and the United Kingdom. In Canada, the Canadian Trusted Computer Product Evaluation Criteria (CTCPEC) version 3.0 was published in early 1993 as a combination of the ITSEC and TCSEC approaches. In June 1993, the sponsoring organizations of the CTCPEC, FC, TCSEC and ITSEC pooled their efforts and began a joint activity to align their separate criteria into a single set of IT security criteria that could be widely used. This activity was named the CC Project. As the outcome of this project "Common Criteria (CC); Evaluation Criteria for Information Technology Security" was produced and recognized by ISO as IS (International Standard) 15408 (Common Criteria Org., 2002a).

The Common Criteria (CC) for Information Technology (IT) Security Evaluation is a standard and basis for specifying and evaluating the security features of computer products and systems especially security products/softwares. It provides a common set of requirements for the security functions of IT products and systems and the assurance measures. The CC supports the development of standardized sets of “well understood” IT product security requirements by user communities in the form of Protection Profiles (PPs) for use in procurements and advice to manufacturers. Manufacturers can use similar sets of CC-based requirements to describe the security capabilities of their products. These are called Security Targets (STs), which can then be used as the basis for security evaluations of those products. Security evaluations are formalized testing and analytic processes that use the CC to determine whether IT products have been correctly developed to specification and whether they are effective in countering the security problems as claimed. Users can integrate evaluated IT products into their systems with increased confidence that their claimed security features will operate as intended (Common Criteria Org., 1999a, 2002b).

### 2.4.1 Parts of the Common Criteria

The CC presents three main parts, called Introduction and General Model, Security Functional Requirements, and Security Assurance Requirements, with distinct

but related content. These there parts of CC are introduced in the following sub-sections. All the information about CC, its background, parts and usage, are compiled from CC Part1 and documents available in Common Criteria official home page (Common Criteria Org., 1999a, 1999d, 2002a, 2002b).

#### 2.4.1.1 Part 1 - Introduction and General Model

Part 1 introduces the CC. It defines general concepts and principles of IT security evaluation and presents a general model of evaluation. Part 1 also defines constructs for expressing IT security objectives, for selecting and defining IT security requirements, and for writing high-level specifications for products and systems. These constructs are called Protection Profiles (PPs), Security Targets (STs) and packages (PP is described in a later section). In addition, Part 1 describes the usefulness of each part of the CC in terms of each of the target audiences.

#### 2.4.1.2 Part 2 - Security Functional Requirements

Part 2 contains a catalogue of well-defined and understood security functional requirements that are intended to be used as a standard way of expressing the security requirements for IT products and systems. The catalogue is organized into classes, families, and components.

- Classes are high-level groupings of families of requirements, all sharing a common security focus (e.g., identification and authentication, auditing).
- Families are lower-level groupings of requirement components, all sharing specific security objectives but differing in rigor or emphasis (e.g., user authentication, audit data generation, audit analysis, audit review).
- Components are the lowest selectable requirements that may be included in PPs, STs, or packages (e.g., unforgeable user authentication).

Part 2 also includes an extensive annex of application notes for applying the material that it contains. While it is possible to explicitly state functional requirements not included in the Part 2 catalog in building CC-based constructs (PPs, STs, and packages), that is not advised unless it is clearly not practical to use Part 2 components. Using functional requirements not part of the catalog could jeopardize widespread acceptance of the result.

#### 2.4.1.3 Part 3 - Security Assurance Requirements

Security assurance requirements are grouped into classes. Classes are the most general grouping of security requirements, and all members of a class share a common focus. Eight assurance classes are contained within Part 3 of the CC. These are as follows:

- Configuration Management.
- Delivery and Operation.
- Development.
- Guidance Documents.
- Life Cycle Support.
- Tests.
- Vulnerability Assessment.
- Assurance Maintenance.

Each of these classes contains a number of families. The requirements within each family share security objectives, but differ in emphasis or rigor. For example, the Development class contains seven families dealing with various aspects of design documentation (e.g. functional specification, high-level design and representation correspondence).

The CC has provided seven predefined assurance packages, on a rising scale of assurance, known as Evaluation Assurance Levels (EALs). These provide balanced groupings of assurance components that are intended to be generally applicable. The seven EALs are as follows:

- EAL1– functionally tested
- EAL2- structurally tested
- EAL3– methodically tested and checked
- EAL4– methodically designed, tested and reviewed
- EAL5– semiformally designed and tested
- EAL6– semiformally verified design and tested
- EAL7– formally verified design and tested

EAL1 is the entry level. Up to EAL4 increasing rigor and detail are introduced, but without introducing significant specialized security engineering techniques. EAL4 can generally be applied to products and systems not developed with evaluation in mind.

Above EAL4, the increasing application of specialized security engineering techniques is required. TOEs meeting the requirements of these levels of assurance will probably have been designed and developed with that objective. At the top level (EAL7) there are significant limitations on the practicability of meeting the requirements, partly due to substantial cost impact on the developer and evaluation

activities, and also because anything other than the simplest of products is likely to be too complex to submit to state of the art techniques for formal analysis.

### 2.4.2 Protection Profile and Common Criteria Usage

The CC is used in two general ways:

- As a standardized way to describe security requirements, e.g., PPs and STs for IT products and systems; and
- As a sound technical basis for evaluating the security features of these products and systems (Common Criteria Org., 1999a)

The CC defines three useful constructs for putting IT security requirements from Parts 2 and 3 together: the PP, the ST, and the package. The CC has been developed around the central notion of using in these constructs, wherever possible, the security requirements in Parts 2 and 3 of the CC, which represent a well-known and understood domain.

The PP is an implementation-independent statement of security needs for a set of IT security products (such as Firewalls, IDSs) that could be built. The PP contains a set of security requirements, preferably taken from the catalogues in Parts 2 and 3, which should include an EAL. A PP is intended to be a reusable definition of product security requirements that are known to be useful and effective.

A PP could be developed by user communities, IT product developers, or other parties interested in defining such a common set of requirements. A PP gives consumers a means of referring to a specific set of security needs and communicating them to manufacturers. The PP also helps future product evaluation against those needs.

The PP contains the following items:

- PP introduction - identification and overview information, which allows users to identify PPs useful to them.
- Target of Evaluation (TOE) description - description of the IT product and its purpose, not necessarily from a security perspective.
- TOE security environment - description of the security aspects of the environment in which the product is intended to be used and the manner in which it is expected to be employed. This statement includes the following:

- Assumptions about the security aspects of the product's expected usage and operating environment, such as value of assets and limitations of use. Assumptions also describe the environment's physical, personnel, and connectivity aspects.
- Threats against which the product or its supporting environment must specifically provide protection.
- Organizational security policies or rules with which the product must comply. These can be any explicit statements of IT security needs that the product must meet.

- Security objectives - a high-level statement of what the product and its environment are intended to accomplish in covering the threats, policies, and assumptions.
- IT security requirements - the detailed statement of IT security functional and assurance requirements that the product and its operating environment must satisfy to meet the objectives.
- Application notes - additional supporting information that may be useful for the construction, evaluation, or use of the product.
- Rationale - the evidence describing how the PP is complete and cohesive and how a product built against it would be effective in meeting the objectives.

**2.5 Methodology**

There are some different ways of organizing the content of software requirement specification suggested by some US governmental and standardization organizations such as Department of Defense DI-MCCR-80025A, NASA's SFW-DID-08 and IEEE/ANSI 830 (Davis, 1990). The following basic requirement titles and content are common mostly for the above standards:

- Functional Requirements: Functional requirements should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs. These are generally listed as "shall" statements starting with "The system shall" (IEEE Std 830, 1998)
  - Functions- description of all operations, functions should be flagged.

- Inputs- definition of inputs: specify sources of inputs, types, formats, units of measure, timing, and ranges of valid inputs, including accuracies and tolerances, disposition of illegal values, error messages.
- Outputs- definition of outputs: specify destinations of outputs, units of measure, timing, range of valid outputs, including accuracies and tolerances, disposition of illegal values, error messages.
- Data Handling- requirements for database or dictionary.

- Security Requirements: Specification of the factors that protect the software from accidental or malicious access, use, modication, destruction, or disclosure. Specific requirements in this area could include the need to
  - Utilize certain cryptographic techniques;
  - Keep log or history data sets;
  - Assign certain functions to different modules;
  - Restrict communications between some areas of the program;
  - Check data integrity for critical variables.
- Performance Requirements: Specification of each performance requirement in testable and quantitative terms. The following should be addressed:
  - Timing and sizing requirements.
  - Sequence and timing of events, including user interaction tolerances.
  - Throughput and capacity requirements.
  - Error detection, isolation, and recovery requirements for data and processes.
  - Fail-safe requirements.
- External Requirements:
  - Interfaces. User, hardware, software, and communications interfaces. The purpose, requirements (e.g., performance, safety, security), and implementation constraints for each interface. For example, requirements for user interface may include screen descriptions and report descriptions.
  - Environment. Operating conditions.

- Assurance Requirements: The assurance requirements may prove specified requirements, functions are achieved by the product. They may include development, documentation, delivery and installation, management requirements

Some parts of above content are included in this thesis also. On the other hand, they are not presented in ordered and title-by-title format as stated above but using developed IDS model and Common Criteria Protection Profile format (which is explained in previous section). The methodology given below is followed for specifying and representing the requirements of IDSs:

- First, a generic IDS model and requirements were tried to be achieved. For this purpose the sub-step specified below were followed:
  - ✓ Available IDS literature and products were searched. Common specifications, requirements and models for IDSs were extracted.
  - ✓ A generic IDS model in client-server architecture was developed which is based on the available literature and products. While developing this model generic IDS needs were tried to be answered.
  - ✓ Individual requirements of the modules were specified on developed model at abstract level. This specification was achieved module by module. Then, the concatenate requirements for IDSs were specified under "System Requirements" title, including the subtitles such as development, installation, configuration.
- An IDS Protection Profile was written specific to PC LAN environment. For this purpose, CC PP development methodology (Common Criteria Org., 1999a; ISO/IEC JTC, 1999), illustrated in Figure 2.3, was followed and previously developed IDS model and requirements in this thesis were also used as the basis for this PP. Below sub-steps were followed in order to develop PP:
  - ✓ First of all, the selected working environment (Izmir Institute of Technology Department of Computer Engineering, PC Laboratory/PcLab) was focused on and its specifications were extracted (assets, physical environment).
  - ✓ Then assumptions and threats for this environment were defined according to its specs.

- ✓ At next step, security objectives ($O_{PG}$ for IDS product group specific for this environment) that are the intended response to assumptions and threats were defined.

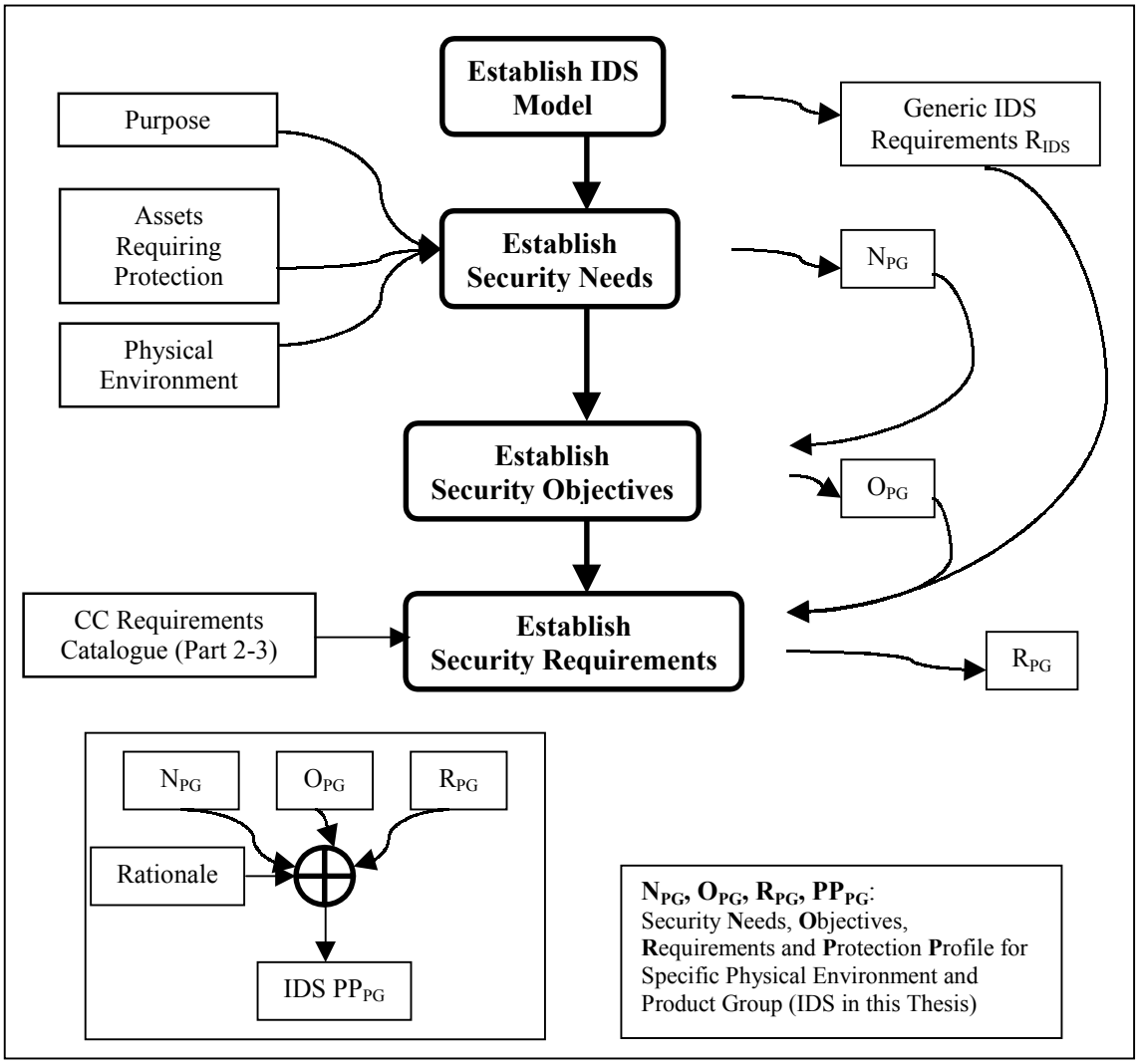


Figure 2.3: IDS Requirement and PP Development Methodology.

- ✓ Security requirements addressing those objectives were written. At this step, CC Part2 and IDS requirements were used. CC Part2 is used as a requirement catalogue, the needed requirements were restated from this catalogue after the allowed modifications. In addition to the available requirement classes in the CC Part 2, a new proposed class was written for intrusion detection systems (see Appendix D). For this purpose, existing CC classes especially Security Audit class (FAU) was used as the model and basis for new written security requirement class. Assurance requirements

including testing, configuration, development requirements were directly retstated from CC Part3.

- ✓ At last, rationale part was added which shows internal consistency between assumptions, threats, objectives and requirements.

At all steps of PP development, other evaluated and validated security product PPs and Security Targets (STs) were used as the formal model such as Firewall-1 Version 4.0 Security Target V2.4. The methodology explained above is illustrated in Figure 2.3 and this methodology is common and applicable to other IT products PP development processes.

# Chapter 3

## GENERIC INTRUSION DETECTION SYSTEM MODEL AND REQUIREMENTS

In this chapter a model of IDS will be introduced that the model helps visualize the functionality of IDS and its components more clearly. This model has a client/server architecture composed of graphical user interface (GUI), management server (MS), database server (DS) and intrusion detection module (IDM) which has two portions called collector and analyzer.

In Figure 3.1, this generic IDS model is presented. According to the model, security administrator controls the IDS through GUI that only communicates with management server and it is not directly connected to the other IDS portions. GUI is the display of intrusion detection system information that appears/is installed on system administrator's computer. All management, configuration, response, reporting and other possible jobs related to management are centralized on management server. Configuration and policies are defined and implemented at a central management server

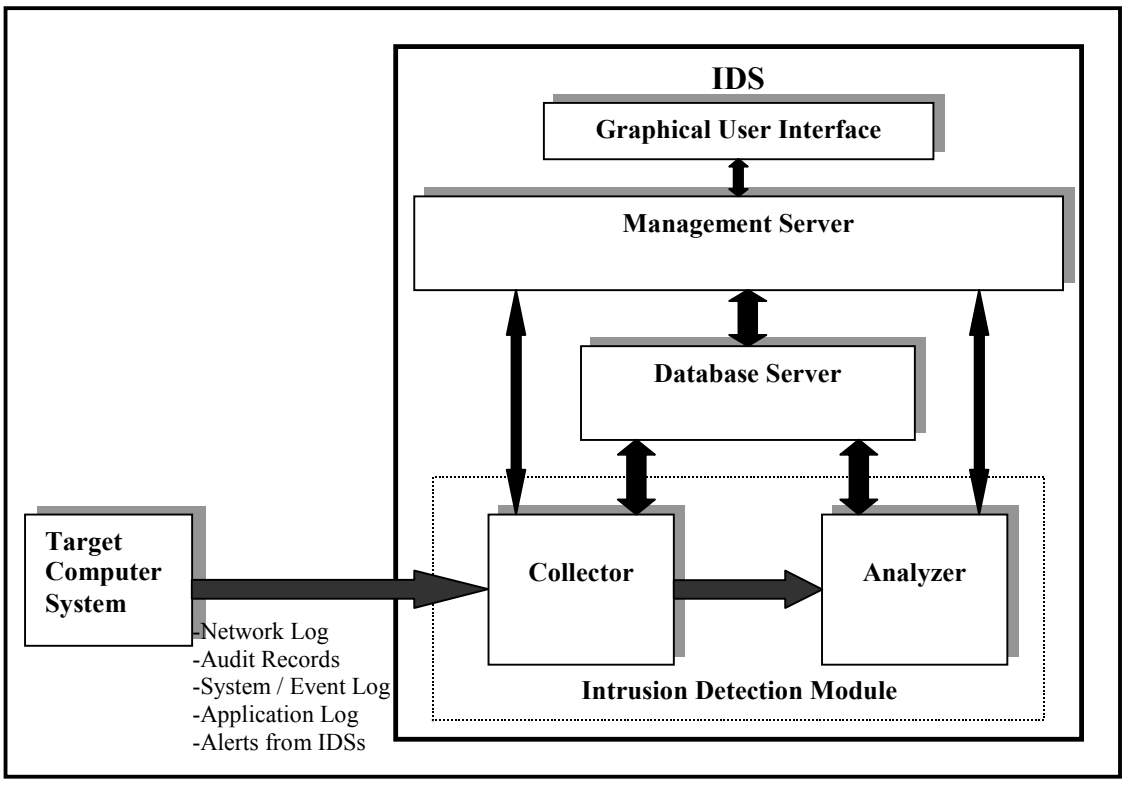


Figure 3.1: Intrusion Detection System Model.

through GUI, each intrusion detection module downloads and applies its individual policy. This centralized client/server architecture allows first, transparency between GUI and other IDS portions (IDM, database) from users' point of view; second, controlling the many remote IDS components from many other remote points by accessing only a single management server. On the other hand, that should be noted, in complex systems many number of management servers can work on different computers in a distributed and synchronous way.

Another IDS component, database server, stores all the short and long-term information available on the IDS. That information may comprise assets in the IDS, IDS policies, configurations, alerts, events, analysis results and collected/pre-processed computer system and network data that can be used by analyzer. Although intrusion detection modules can reach the database directly, a user only sees the information presented by management server. For instance, when a user wants to get an event report from GUI, GUI connects to the management server and handles this request. Management server processes the request and produces the report from the data in the database, finally it sends the output back to the GUI.

As mentioned before, the assets' information, policies and configurations of the assets are also stored in the database, but intrusion detection modules can not reach their policies directly. When any policy or any configuration is changed/modified by the administrator, IDM sends the "update" signal to all related modules, once they get the "update" signal then they can download their necessary information through management server. This architecture may have some benefits. For example, in this way intrusion detection modules can be grouped according to their specifications and management server can supply the group policy in which intrusion detection module belongs to. This can make management relatively easier.

The last part of the model is intrusion detection module, which consists of two sub-modules. The first one, named collector, collects the necessary data from the target computer system or network. These data can be both pre-processed data by another security tool or the raw data such as network packets, audit records, computer system / event log, application log and so on. According to data collected they will be filtered, processed and put into a common format that analyzer can understand. Again according to the data type, collected and pre-processed data can be either sent to analyzer for short-term analysis or stored in the database for long-term analysis or both.

The second one, called analyzer, gets the collected data from either database or collector(s) directly and performs various types of analysis such as statistical and rule-based methods to detect any possible intrusion. When it detects an intrusion, it sends the pre-defined alert to management console and also stores the pre-defined attributes of detected event on the database.

After above overview of the model, in the following subsections the IDS components' requirements will be introduced.

### 3.1 Graphical User Interface

GUI addresses the type of interface provided to the intrusion detection system administrator for configuration, monitoring and administrating both IDS and computer system. The user interface is a key feature of any intrusion detection system, as it provides the administrator with a window into the system for configuration, operation, and maintenance (Amoroso and Kwapniewski, 1998).

- All necessary operations should be performed from within [preferably multiple] user interfaces from which an administrator can remotely control, operate and view any component of the intrusion detection system and any available information on the system by connecting and using the management server functionalities as a client.
- Other IDS components should continue to work if the system's user interface is disconnected, disabled, or experiences denial of service (Lodin, 1999).
- Any tools that can reasonably operate in background mode should be transparent to the user to the extent possible (LaPadula, 1997).
- GUI should be able to graphically display the following things and should allow configuring them: Modules available in the IDS, management window, reports, detected intrusions, policies, configuration options and optionally the other components of computer system and network such as firewalls or switches.

### 3.2 Management Server

All administrative and management functionalities of IDS are centralized at Management Server which is equivalent to middle tier in three-tier client server architecture. The requirements of Management Server are detailed in this section.

- Management server (MS) controls all the other IDS components. This control includes adding new components, starting and stopping the components, changing their configuration and other possible operations done on the IDS.
- MS supplies policies and configurations to the other IDS components. When any change/modification occurs related to IDS modules it sends notification signal to the related modules and they update this change.
- MS answers the connection and service requests of the IDS components and allows the connection if the component is in the asset database (For the details of asset database see database server section 3.3.).
- MS should allow configuration of the IDS via graphical user interface to perform intrusion detection based on the misuse detection, anomaly intrusion detection model or any other technique used in IDS (Metcalf and LaPadula, 2000).
- MS should be able to supply all the information in the database to the GUI.
- MS should have human-understandable report generation capability based on the data stored in the database (Barber, 2001).
- Reports may include description of suspected intrusions based on recorded intrusion data, percentages of the intrusion types and points, network loads, and any other data can be obtained from database selectively.
- Reports should be flexible, extensible, printable, configurable and exportable to different formats, such as text or html.
- MS gets all intrusion alarms coming from the intrusion detection modules available in the asset database. Depending on the policy, it reacts to the intrusion and it sends the display information when any GUI connects to itself.
- The reaction could be the alerts graded according to their seriousness. These alerts should have capability to be activated via different notification channels to different alert destinations, such as e-mail to administrator, alert lines on user-interface and reports. All recorded information about the intrusion should be included in the alert.
- Responses to events should include the ability to deny, degrade, disturb and deceive the intruder, to reconfigure computer systems and information functions and to counterattack. MS should have the capability to initiate those responses automatically such as reporting, closure of session, resetting the connection. Those

automated responses should be selected from predetermined list of responses (Metcalf and LaPadula, 2000) or created by administrator.

### 3.3 Database Server

As the third tier of client-server model the database server provides storage functionality to the intrusion detection systems. Although the third party developers supply database servers in general, the database server and its requirements are stated as a part of generic IDS model.

- Database server (DS) should allow connection requests from the pre-defined intrusion detection modules and management server with authentication.
- DS store the information about the IDS components/assets (asset database).
- DS should store the data collected by the collector for defined period of time (audit database).
- DS should store intrusion information produced by analyzer for defined period of time (intrusion database).
- DS should store the policies (policy database).
- The database should have sorting, querying, and backup capabilities.
- The database ideally should be a commercially available off-the-shelf system (LaPadula, 1997).
- DS should send an alarm if the storage capacity has been reached (Science Applications International Corporation (SAIC), 2000).
- Exporting the data to external database capability should exist (Lodin, 1999).
- Asset database should store at least the following attributes of IDS modules:

Table 3.1: Asset Database Table (cont. on next page).

| **Attributes** | **Description** |
|---|---|
| Module ID | Identification number. |
| Module name | Optional, user-friendly name. |
| Module type | A unique name or number defining the type of module. |
| Host name or IP address | Host name or IP address of the host on which IDS module is installed. |
| Policy ID | Identification number of the policy applied. |
| Control port number | Used by MS for controlling the module. |

Table 3.1: Asset Database Table (cont.).

| Attributes | Description |
|---|---|
| Communication port | Pot number used by analyzer and collector to communicate with each other. |
| Database port number | Port number used for communicating with database. |

## 3.4 Intrusion Detection Module

Detection job is performed by Intrusion Detection Module. This module includes two main sub-systems called, Collector and Analyzer. In this section, generic functional requirements of these modules are defined.

- An intrusion detection module should include at least a collector and an analyzer or a single module doing their jobs specified below.
- IDM should connect and download its policy from management server periodically and/or when an update directive comes from MS.

### 3.4.1 Collector

- Collector should be able to collect the specified information from the targeted computer system resource(s) or network for specified period of time according to its type and policy.
- There can be various types of collectors, which are able to collect information from multiple operating systems, networks and multiple platforms (hosts, switches, routers, etc.) at several locations within the network (LaPadula, 1997; Metcalf and LaPadula, 2000).
- Collector should process the collected information. Then it should store them in the database or send to the analyzer or both. Processing stage should involve filtering the data and converting it into the format that is understandable by the analyzer.
- The collected data may include start-up and shutdown, identification and authentication events, data accesses, service requests, network traffic and other specifically defined events such as alerts generated by other IDSs. The details should be collected are shown in Table 3.2. Default attributes for the events are as follows; Event ID, Date and time of the event, type of event, subject identity, and the outcome (success or failure). When host-based events are collected, for the identification and authentication event, the source address could be a subject IDS on a local machine and the destination is defined by default. For the data access and

data introduction events, the source address could be file name and the destination address may be target location for the file

Table 3.2: Events Table (SAIC, 2000).

| **Event** | **Additional Attributes of the Event** |
|---|---|
| Start-up and shutdown | None |
| Identification and authentication requests | User identity,<br>location,<br>source address,<br>destination address |
| Data accesses | Object IDS,<br>requested access,<br>source address,<br>destination address |
| Service requests | Specific service,<br>source address,<br>destination address |
| Network traffic | Protocol,<br>source address,<br>destination address,<br>source port, destination port,<br>data portion of the packet |
| Security configuration changes | Source address,<br>destination address |
| New data creation | Object IDS,<br>location of object,<br>source address,<br>destination address |
| Audit function start-up and shutdown | None |

- According to intrusion detection technique, additional tuples/attributes can be collected for the events, such as resource usage, CPU time or I/O units used (Internet Security Systems, 1999).

### 3.4.2 Analyzer

- Analyzer should get data, which will be analyzed, from the DS or the collector.
- Analyzer should perform at least one of the following analysis methods on supplied data: anomaly detection, misuse detection, integrity checking and other analytical method according to its type and policy.
  - Anomaly analysis involves identifying deviations from normal patterns of behavior. For example, it may involve mean frequencies and measures of variability to identify abnormal usage.

- Misuse analysis involves the use of patterns corresponding to known attacks or misuses of a computer system. For example, patterns of computer system settings and user activity can be compared against a database of known attacks.
- Integrity analysis involves comparing computer system settings or user activity at some point in time with those of another point in time to detect unexpected differences.

- Analyzer should respond when intrusion is discovered. It sends an alarm to the management server and stores the data in database.
- Analyzer should store the data shown in Table 3.3 about the detected intrusion in the intrusion database. The other available data about the event (shown in Table 3.2) should be added to Table 3.3.

Table 3.3: Intrusion Database Table.

| **Attribute** | **Description** |
|---|---|
| Intrusion ID | |
| Timestamp | Detection date and time |
| Intrusion type | This can be an ID number if an intrusion type database exists. Otherwise, the name of the intrusion type and a short description of intrusion. |
| Priority | Shows the importance of detected intrusion. |
| Event ID | The ID number of the last event that caused this intrusion alert. |
| Module ID | ID of the module, which detects the intrusion. |

## 3.5 System Requirements

IDS requirements which, are common for all modules of IDS model, are represented in this section. These requirements have following sub-groups, development; installation, deployment and update capability; configuration; performance; testing and documentation. Most of the requirements in this section are valid for and applicable to not only IDSs but also other software products.

### 3.5.1 Development

There are some common standardized requirements on software development, which are valid for not only IDS but also all the software products. In this section, only some basic software development related requirements are listed.

- The developer should provide functional specification and the high-level design of the IDS (Common Criteria Org., 1999c).

- The high-level design should describe the structure of the IDS in terms of modules and security functionality and functional specification of each module as being done in the IDS model.
- Software verification and validation (V&V) activities should be followed and documented which ensures that IDS satisfies functional and other requirements.
- V&V activities shall be performed during each phase of the software life cycle and shall include analysis of system and software requirements allocation, verifiability, testability, completeness, and consistency (NASA-STD-2201-93, 1992).
- Process and product standards, which lead to high quality, should be defined or selected. These standards should be followed and applied by the software development team. Product standards define characteristics that all product components should exhibit; process standards define how the software development process should be conducted (IEEE, 1999; NASA-STD-2201-93, 1992; Schultz, 2000; Sommerville, 1996). Basic software engineering standards and their coverage are presented in the appendix.
- The developer should provide a security policy model for the IDS and should demonstrate correspondence between the functional specification and the security policy model (Common Criteria Org., 1999c).

### 3.5.2 Installation, Deployment and Update Capability

This section denotes criteria about the installation, deployment and updates of the IDS.

- IDS should support generally used existing operating systems such as Windows2000, Linux or Solaris, computer system and network infrastructure such as Ethernet (Lodin, 1999).
- IDS should be installed quickly and easily without any delay or central points of failure to the network or host being monitored.
- IDS should be scalable such that IDS or its modules should be able to be installed on several number of hosts. This group of hosts should work as a single logical IDS. This structure should allow the balance and distribution of workload and error handling (ISA Server2000, 2000).

- IDS should warn the mis-installation that can cause a system security hole or low performance (ISA Server2000, 2000).
- IDS should be able to be updated with new intrusion signatures or any other product updates automatically or manually via any secure way such as a secure encrypted internet connection or a CD-ROM (Dragon 5, 2002, Norton Antivirus, 2002).
- The IDS should remind periodical updates automatically (Norton Antivirus, 2002).
- Vendor should notify the security officer about the availability of new non-periodical updates (Internet Security Systems, 2000).

### 3.5.3 Configuration

The underlying working environment of IDSs can change from organization to organization and/or from day to day in the same organization as well. In addition to that, new threats are appearing every day. That's why configuration has an important role for adapting the changing work-environment and increasing the performance of IDSs. An assortment of requirements addressing the configuration is stated in this section.

- IDS should allow at least configuration of below items and any other specification improving the detection and usage capability of IDS.
  - Detection technique: Selecting the intrusion detection technique to be applied.
  - Numerical values: Tuning detection threshold levels or any other numerical detection values according to detection technique.
  - Detection rules: Selecting and creating attack signatures.
  - Responses: Selecting or creating the responses for detected intrusions.
  - Managed modules: Adding and removing managed modules by the MS. Defining the locations of the modules.
  - Database: Adding and removing users, defining database roles, deleting certain or all attributes in the tables, clearing the database.
  - Data: Changing the content of the collected and stored data.
  - Policy: Selecting, creating and applying the security policies.
  - Reports: Changing the format and including of reports.
  - Ports: Specifying the communication ports of modules.

- Paths: Specifying paths for reports, configurations, policies, encryption keys and any other needed file.
- Encryption options: Selecting the encryption technique and specifying the key.
- Update: Selecting, enabling and disabling the update way.

- IDS should have configuration and policy templates that guide the user. These templates should be customized and applied by the security officer (Internet Security Systems, 2000).
- IDS should warn mis-configurations that can cause a system security hole or low performance (ISA Server2000, 2000).

### 3.5.4 Performance

The primary performance metric for an IDS is the intrusion detection rate according to most of the studies (Durst, et al. 1999; Lippmann, et al. 1998; Lippmann, et al. 1999; Puketza, et al. 1994). This metric can be defined as the ratio between number of detected intrusions and the number of intrusions, which can be counted in a test-bed laboratory environment.

However, intrusion detection rate is insufficient when considered alone. It must be combined with false alarm rates for normal traffic. False alarm rate can be defined as the number of false intrusion alarms for a period of time such as a day. High false alarm rates such as above hundreds per day, make an IDS excessively unusable, even with high intrusion detection rate because, false alarms require assessment and dismissing operation done by system security officer and this operation increases the human workload. Low false alarm rates combined with high detection rates are desired, and mean that alerts and so IDS can be trusted.

Other common software performance metrics are also considerable such as traffic handling capacity, run-time memory and CPU usage (Amoroso and Kwapniewski, 1998; Durst, et al. 1999; Lippmann, et al. 1998; Lippmann, et al. 1999; Puketza, et al. 1994; Puketza, et al. 1995).

These metrics can have different relative values and importance, at different computer systems and networks under different working conditions. For example, a broad detection range may not be necessary if the IDS monitors a site that is protected by other security mechanisms and tools (Puketza, et al. 1994) or false alarm rates may

be lower, if a firewall might block most network traffic, and reduce the load on the network. Economy in resource usage may not be required at a computer system and network where security has a high priority and where computing resources exceed user needs. Resilience to increasing load may be less important for a computer system and network where users can not monopolize resources and not increase system load. Thus, importance of the performance requirements is relative; they may change and be identified accordingly.

- IDS should function without using too much system resource such as main memory, CPU time disk space (Puketza, et al. 1994).
- IDS's detection capabilities should not be diminished as load increased (Barber, 2001). Load increase can be the increase in the number of packets travelling on the network for a time interval, the number of hosts to be observed or the number of users on a host. Detection capabilities can be defined in terms of false alarms per day, detection rates.
- Increasing the number of attack signatures should not significantly impact the detection capabilities of IDS (Internet Security Systems, 2000).
- Collecting the information, monitoring, scanning the system should not have noticeable effect on normal the computer system and network operations (Metcalf and LaPadula, 2000).
- IDS should minimize detection and reaction time required for real-time systems.

### 3.5.5 Testing

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design, and coding (Pressman, 1994). Testing of a software starts from development phase such as low-level tests that are necessary to verify that even a small source code segment has been correctly implemented; to the end product such as high-level tests that validate major software functional requirements. Software testing strategies, methodologies and metrics are detailly explained in many number of researches and books (Information Processing Ltd, 1996a; Information Processing Ltd, 1996b; Pointe Technology Group, Inc., 2000; Pressman, 1994). In this section, basic high-level test requirements and metrics are introduced for an IDS.

- IDS should be tested by its developer.
- Requirements defined in this chapter, all the requirements and specifications of the product, and additionally other possible common software requirements such as reliability, maintainability, portability, correctness, completeness, easy of use and simplicity (Williams et al. 1998) should be checked during the tests even starting from the delivery and installation.
- The performance metrics and requirements specified in section 3.5.4 should be tested and documented. These metrics should include at least the system resource usage, intrusion detection rates, and false alarms rates under different levels of workload.
- The testing methodology can be based on simulating the real computer system and network with users -both intruders as well as normal users while IDS is running. In the simulation, IDS should be tested by the simulated attacks to the computers in the test bed/laboratory under the simulated working conditions near to real-world conditions (Amoroso and Kwapniewski, 1998; Lippmann, et al. 1998; Lippmann, et al. 1999; Puketza, et al. 1994).
- Following possible reliability parameters should be tested (Pressman, 1994; Sommerville, 1996):
    - Availability: Specify the percentage of time available, hours of use, maintenance access, degraded mode operations, and so on.
    - Mean Time Between Failures: This can be specified in hours, days or months.
    - Mean Time to Repair: Out of operation time after the fail.
    - Maximum Bugs or Defect Rate: Expressed in terms of bugs per thousand of lines of code (bugs/KLOC) or bugs per function-point(bugs/function-point).
    - Bugs or Defect Rate: Categorized in terms of minor, significant, and critical bugs. For example, a “critical” bug can be complete loss of data or a complete inability to use certain parts of the system’s functionality.
- The test results should show the anticipated outputs from a successful execution of the tests. The test results should demonstrate that each function operates as specified (Common Criteria Org., 1999c).

### 3.5.6 Documentation

Documentation may include many kinds of documents such as reports, development products, delivery procedures, and user guides. Complexity, architecture and some features of a product may affect the amount of documentation a product should need. Also some documents may be presented together as a single document such as installation, guidance and configuration. “For example, the design documentation may consist of a single document describing both the system architecture and the detailed modules or it may consist of separate documents for the architecture and sub-systems” (Wallace et al 1994). The purpose of this section is to identify the basic content and requirements of documentation needed for an IDS.

- The developer should provide at least the documents below in both hard and soft media.
  - Development: Describing high-level design with functional specifications.
  - Delivery: Describing all procedures to maintain secure distribution of the IDS, its updates or other deliveries to the user.
  - Installation: Describing procedures for the installation, generation, and start-up of the IDS.
  - Guidance: Describing all modules functions, modes of operation, specifications and usage.
  - Intrusions: Describing the known intrusions, including their name, type, effect, method and other possible information about the intrusion.
  - Configuration: Describing how IDS is configured and configuration specifications.
  - Testing: Describing test plans, test procedure & methodology descriptions, expected test results and actual test results.
- The documentation should be complete, clear, consistent and well organized (Lodin, 1999).
- The documentation format should fit the common documentation standards developed by IEEE or ISO (see Appendix B for documents covered by each organization).

- On-line and off-line helps that are integrated with GUI should be supplied. Help should include all the installation, guidance and configuration items defined above.

### 3.5.7 Security

These requirements focus on the degree to which the intrusion detection system protects itself from malicious security attack. Techniques such as encryption and access control are typical controls used to ensure the security given below (Amoroso and Kwapniewski, 1998):

- All modules of IDS should ensure trusted communications between themselves; for example, an analyzer should authenticate itself to the management server and vice versa (Metcalf and LaPadula, 2000).
- All communication between the IDS modules should be encrypted (Amoroso and Kwapniewski, 1998; Lodin, 1999; Metcalf and LaPadula, 2000; Dragon 5, 2002)
- IDS should require user authentication before allowing any operation.
- IDS should log the operations performed.
- IDS should monitor its own activities for signs of interference, failure or intrusions, and is capable of responding when such signs are found (Jackson, 1999).
- IDS should be capable to check and verify the integrity of the transmitted and stored data by using digital signatures, MD5 checksum or any other way (Jackson, 1999; NetSonar100, 2001).
- All security mechanisms of IDS should be self-contained and does not require pre-existing security infrastructure such as public key infra structure (HP IDS/9000, 2001).

# Chapter 4

# INTRUSION DETECTION SYSTEM PROTECTION PROFILE FOR PC LABORATORY LAN ENVIRONMENT

## 4.1 Protection Profile Introduction

This chapter of thesis is written in the Common Criteria (CC) Protection Profile (PP) format, on the other hand in some sections some additional explanatory parts are added for the reader who is not familiar with CC.

This introductory section presents PP identification information and a brief discussion of the PP.

This PP is developed for an Intrusion Detection System for PC Laboratory LAN Environment and it can be stated as Target of Evaluation (TOE) in the rest of this chapter according to CC terminology.

### 4.1.1 Identification

**Title**: Intrusion Detection System Protection Profile for PC Laboratory LAN Environment

**Keywords:** intrusion detection, network, security, protection profile, LAN.

### 4.1.2 Overview

*Following topics should be covered in this Overview section of PP: the targeted IDS, and PC Laboratory LAN environment, function of this IDS, its structure and working principles overview in abstract level. The developed and previously mentioned IDS model can be summarized and restated directly here.*

## 4.2 Target of Evaluation Description

### 4.2.1 Architecture and Working Principles

*Modules in the IDS model, their functions, and interactions should be explained here. Target of Evaluation is explained again in much more detail than the PP Overview section. IDS Model can be used in this section too.*

### 4.2.2 Scope and Boundaries of the Target of Evaluation

This section provides a general description of the physical and logical scope and

boundaries of the TOE.

The TOE configuration consists of one physical component/host including for PcLab LAN environment:

- One computer with operating system and database server, which are integrated with TOE.
- Two network interfaces.
- One graphical user interface (GUI), one management server (MS), and one intrusion detection module (IDM) which has three portions two collectors and one analyzer.

Software and hardware features outside the scope of the defined TOE Security Functions (TSF) are as follows: client and session authentication between modules, interaction with other products such as database servers, and remote administration. These are out of the scope of TOE because the TOE is not maintained in a distributed way in PcLab environment.

### 4.3 Target of Evaluation Security Environment

The IDS PP is prepared for the LAN environment, which is similar to the İzmir Institute of Technology, Department of Computer Engineering, PC Laboratory (PcLab). All assumptions, threats and objectives stated in the IDS PP rely on this PcLab LAN environment. Therefore, the following explanatory part about the PcLab is needed to make the PP clearer.

This PcLab is serving to 100's of computer engineering students who are using common applications and services including but not limited to, sending and receiving e-mails, using FTP to send and receive files, accessing other computers via telnet sessions, browsing web pages, and compiling and running the programs. These user actions can be simply based on the following services and protocols: http, smtp, pop3, ftp, irc, telnet, X, SQL/telnet, dns, finger, and snmp.

In this laboratory, there are 32 computers with Win2000, Win98 and Linux operating systems inter-connected through an ethernet network (Figure 4.1). There are 29 computers for students' usage. These computers are indicated as Pc(1:n) in Figure 4.1. All Win98 machines are working stand-alone so they do not make any password authentication from any server. On the other hand, Linux machines are using NIS and

NFS services for authentication and mounting the home directories of the users from Linux Server. So all Linux users are defined on the Linux Server. Client Linux machines get user specific information including groups, passwords, and settings from Linux Server. Student PCs are connected to the internet via Linux Server and then through Firewall. So while the Linux Server is the gateway for the student PCs, Firewall is the gateway for all computers in the lab.

Network based IDS is sniffing network packets from both hubs in the lab in order to detect intrusions from both internal and external networks.

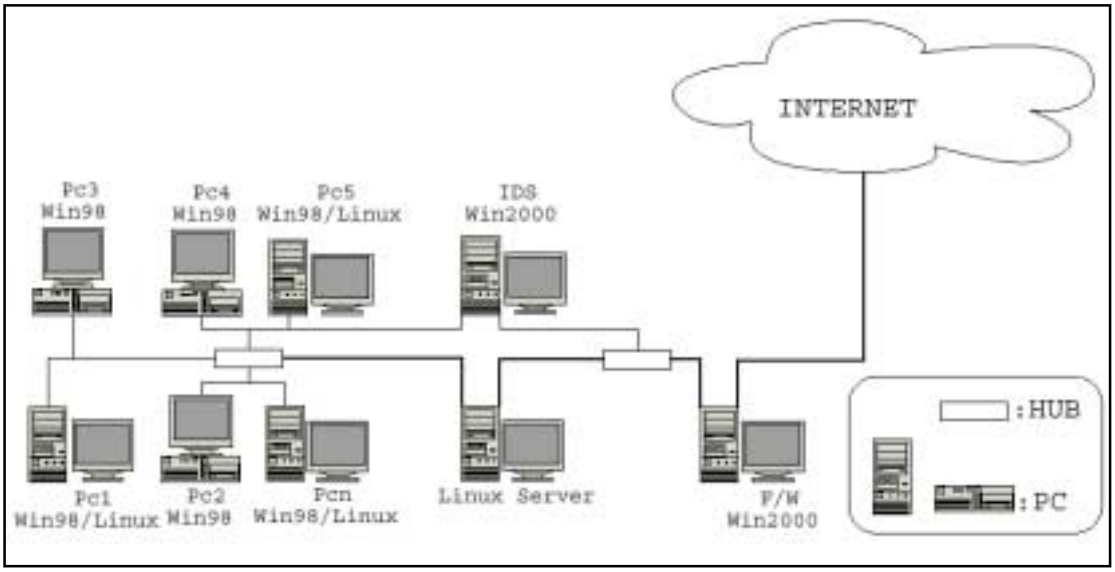


Figure 4.1: PcLab Topology.

IDS, Firewall and Linux Server machines have dual ethernet cards as a result of jobs they are performing. The other specifications of computers available in PC Lab are shown in Table 4.1.

Table 4.1: PcLab Specification Summary (cont. on next page).

| | |
|---|---|
| **Student PCs** | Function: Student usage.<br>O/S: Redhat 7.0 Linux and Win98<br>Total number: 29<br>Supplied Services: Common user applications such as compilers, browsers, telnet, ftp and e-mail programs.<br>Physical Access: Only Computer Eng. Students. |

Table 4.1: PcLab Specification Summary (cont.).

| | |
|---|---|
| **Linux Server** | Function: Server, gateway.<br>O/S: Redhat 7.2 Linux<br>IP: Dual Ethernet; (Student PCs side, Firewall side)<br>Supplied Services:<br>WEB (for experimental purposes),<br>FTP (anonymous access enabled),<br>TELNET,<br>NIS,<br>NFS,<br>SAMBA servers,<br>Common user applications.<br>Physical Access: Only administrator. |
| **IDS** | Function: Intrusion detection.<br>O/S: Win2000 (only administrator, no user account)<br>IP: Dual Ethernet (Student PCs side, Firewall side)<br>Supplied Services:<br>Only O/S and IDS services,<br>No any file or source sharing.<br>Physical Access: Only administrator. |
| **Firewall** | Function: Firewall, Gateway.<br>O/S: Win2000.<br>IP: Dual Ethernet (Internet side, Linux Server side).<br>Supplied Services:<br>Firewall management,<br>Windows network file sharing,<br>TELNET,<br>FTP (anonymous access enabled).<br>Physical Access: Only administrator. |

The laboratory is administered by computer engineering department research assistants who know all administrative passwords in the laboratory. Those research assistants also control the physical access to the laboratory during the office hours (09:00-17:00). The laboratory is locked out of the office hours. On the other hand, students can study at PcLab out of the office hours by taking the permission from research assistants. In that case, the permitted students can study at the lab without any administration and the names of these students are logged according to date. The PcLab is locked again by Security Personnel at the end of this extra studying period. The

Security Personnel is available at the entrance of the building for 24 hours and this person is responsible for keeping the lab doors locked out of the office hours.

As seen from the above explanations about PcLab, this laboratory has common specifications for LAN environments so this PP could be applicable to similar LAN environments.

### 4.3.1 Assumptions

In this section, following types of assumptions are included that are valid for PcLab LAN operational environment:

- Physical protection of any part of the TOE.
- Personnel aspects (e.g. the types of user roles anticipated, their general responsibilities, and the degree of trust assumed to be placed in those users).
- Connectivity aspects (e.g. a firewall being configured as the only network connection between a private network and a hostile network).
- Underlying system aspects.

Almost all the assumptions are quoted from Traffic-Filter Firewall PP (Department of Defense [DOD], 2000) and modified according to existing PcLab environment conditions.

#### 4.3.1.1 Physical Assumptions

**A.PROTECT** The TOE is assumed to be physically protected from unauthorised modification by potentially hostile outsiders.

#### 4.3.1.2 Personnel Assumptions

**A.ADMIN** It is assumed that one or more non-hostile authorised administrators are assigned who are competent to manage the TOE and the security of the information it contains, and who can be trusted not to deliberately abuse their privileges so as to undermine security and follow all administrator guidance; however, they are capable of error.

**A.ATTACK** Attackers are assumed to have a high level of resources and motivation.

**A.NOREMO** Human users can not access the TOE remotely from the internal or external networks.

#### 4.3.1.3 Connectivity Assumptions

**A.IDS** IDS is assumed to be configured and maintained as it can access to and collect all network packets it needs to perform its functions.

**A.IDSNET** IDS is maintained as no information can flow in the internal networks without reaching to the IDS also.

#### 4.3.1.4 Underlying System Assumptions

**A.GENPUR** There are no general-purpose computing capabilities (e.g., the ability to execute arbitrary code or applications) and storage repository capabilities on the TOE.

**A.PUBLIC** The TOE does not host public data.

**A.TOECONF** The TOE is installed, configured, and managed in accordance with its evaluated configuration.

**A.AUDIT** The underlying operating system will audit the actions of its users.

**A.ALLINS** All TOE modules are installed on the same computer.

**A.DATABS** The database as an essential part of IDS is supplied by a third party and it should be evaluated separately from the IDS.

### 4.3.2 Threats

Threats to the assets in a PcLab LAN environment composed of Linux, Win98 and Win2000 operating systems will be introduced in this chapter. These threats could be both to the IDS or its security environment.

In order to identify the threats, the following questions are needed to answer (ISO/IEC JTC, 1999):

a) What are the assets that require protection?

b) Who or what are the threat agents?

c) What attack methods or undesirable events do the assets need to be protected from?

These questions can be answered either in both top-down or bottom-to-up approach. In the first approach, the assets of the IT system are defined first and then possible threats and attackers to those assets are introduced.

In latter approach, the attack methods are defined first and then the assets affected by those attacks are introduced. While defining the threats against to IDS and its security environment, the second bottom-to-top approach is followed in this thesis.

For this reason, Kendall's (Kendall, 1999) attack description model is used, which is presented in Figure 4.2. This model represents five main methods to be followed by the attacker and some possible actions done at the end or during these methods. In Kendall's model, an attacker has an initial privilege level; the attacker may get the higher privileges than the initial state or stay at the same state at the end of the attack and does some action such as denying a service. Kendall's model may be used as illustrated in the following examples:

**R-a-Deny(temporary):** A user with remote network access temporarily denies a service.

**U-b-S-Alter(Files):** An attacker with a local account uses a bug in a program to gain root access and alter files.

Single letters are used to represent the privileges and methods in Figure 4.2. Words in the paranthesis after action part show the target or the type of the action.

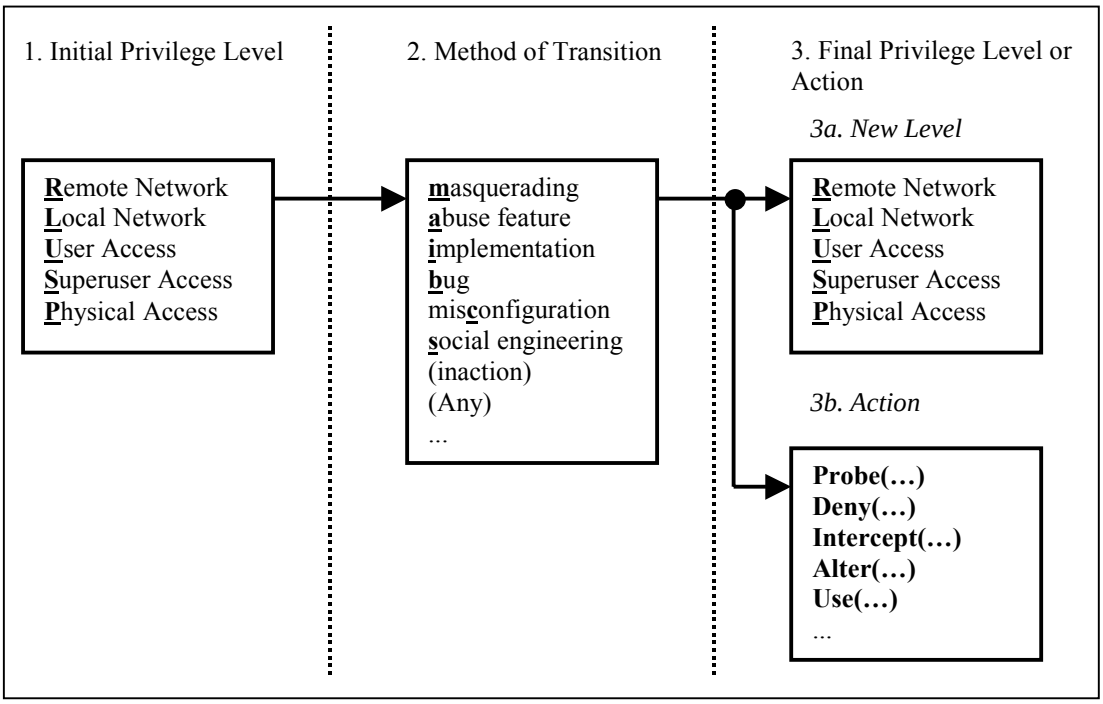


Figure 4.2: A Summary of Possible Attack Description (Kendall, 1999).

In the following part possible threats will be introduced. These threats are produced and grouped by using an attack method oriented view, based on the model

above. However this approach is aiming to define the attacks; that's why it is insufficient to define all the threats. In order to make it sufficient, some more methods can be added in the model such as misuse resulting from inaction or inaction in short form. These addings, which are not included Kendall's original model, are written in parenthesis in the Figure 4.2.

For simplification of referencing the threats, they are uniquely labelled by "T.", indicating that it is a threat, followed by the abbreviation of individual threat (eg., T.NOAUTH).

#### 4.3.2.1 Threats Countered by the Target of Evaluation

In this sub-section the threats countered by the IDS are defined. These threats are mainly divided into two groups according to their target: threats targeting the IDS and threats targeting the IT system which IDS is monitoring. In Figure 4.3 threats targeting the IDS are illustrated and grouped according to their subjects initial privilege level, method of transition and final state. The details of those illustrated and grouped threats are stated in Table 4.2. First three column of Table 4.2 and 4.3 specify the threat according to Kendall's (Kendall, 1999) attack description model, fourth column includes the abbreviation of threat and at last column the details of each threat are specified.

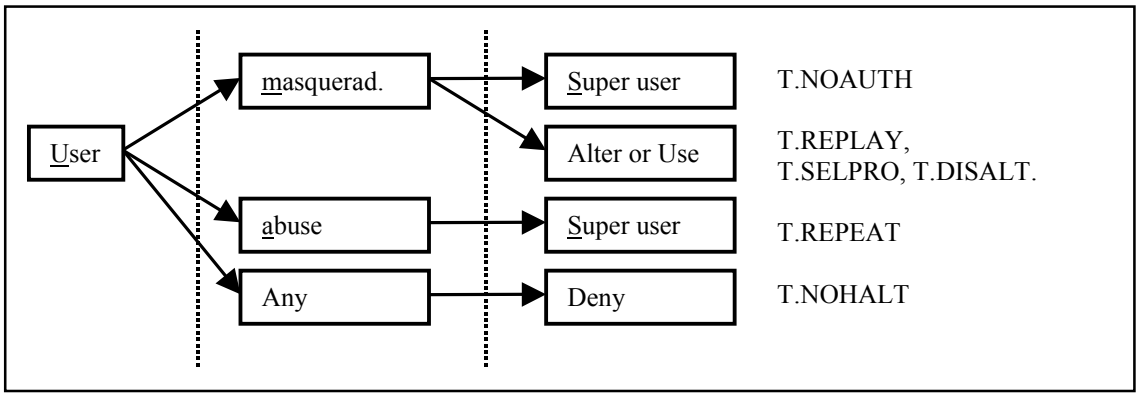


Figure 4.3: TOE Threats.

Table 4.2: TOE Threats.

| Initial Privilege Level | Method of Transition | New Level or Action | Threat Name | Threat Description |
|---|---|---|---|---|
| U | m | S | T.NOAUTH | An unauthorised user may attempt to bypass the security of the TOE so as to access and use security functions and/or non-security functions provided by the TOE. |
| U | a | S | T.REPEAT | An unauthorised person may repeatedly try to guess authentication on the TOE. |
| U | m | Alter or Use | T.REPLAY | An unauthorised person may use valid identification and authentication data obtained to access and use functions provided by the TOE. |
| | | | T.SELPRO | An unauthorised person may read, modify, or destroy security critical TOE configuration data. |
| | | | T.DISALT | An unauthorised user may attempt to disclose and alter the data collected and produced by the TOE. |
| U | Any | Deny | T.NOHALT | An unauthorised user may attempt to deny the TOE's functions such as collection and analysis functions by halting execution of the TOE. |

In Figure 4.4 threats targeting the IT System are illustrated and grouped according to their subjects initial privilege level, method of transition and final state.

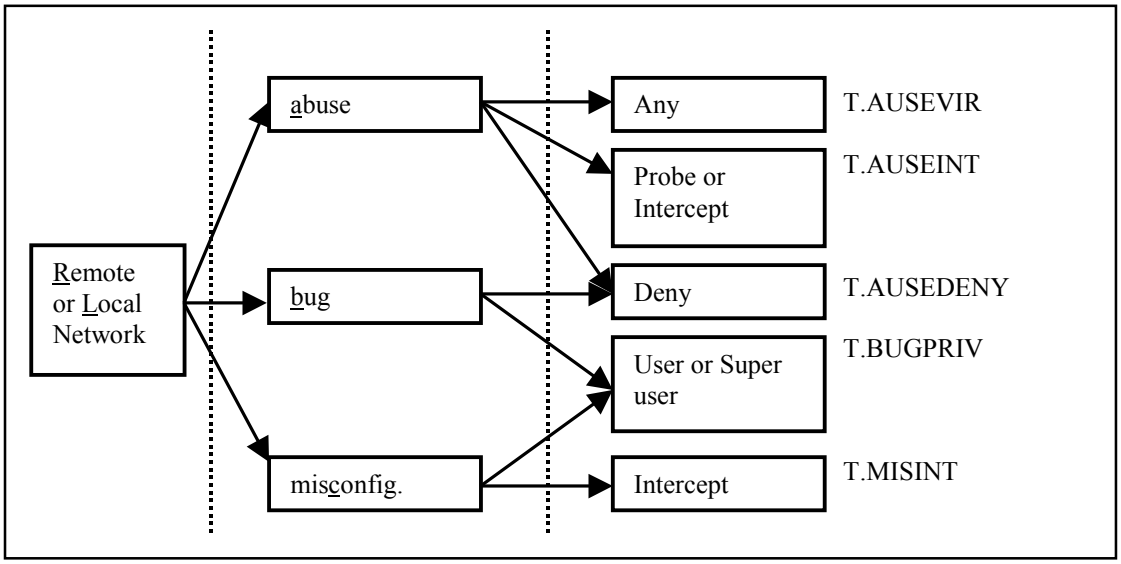


Figure 4.4: IT System Threats.

Then threats targeting the IT System are represented in Table 4.3 with their definitions.

Table 4.3: IT System Threats.

| Initial Privilege Level | Method of Transition | New Level or Action | Threat Name | Threat Description |
|---|---|---|---|---|
| R or L | A | Any | T.AUSEVIR | An attacker from local or remote network may abuse the network to send any malicious code such as virus or trojan horses to the IT system. |
| R or L | a or b | Deny | T.AUSEDENY | An attacker from local or remote network may abuse a system feature or use a bug of a system service to deny the service in the IT system such as FTP, TELNET, and NFS. |
| R or L | b or c | U or S | T.BUGPRIV | An attacker from the remote or local network may use a service bug, or misconfiguration in the TOE Environment to get access to the hosts and to have higher privileges than he/she already has on the accessed host in the IT system. |
| R or L | C | Intercept | T.MISINT | An attacker from the remote or local network may use a misconfigured service in the TOE Environment to intercept the unauthorised information from IT system such as key strokes. |
| R or L | A | Probe or Intercept | T.AUSEINT | An attacker from the remote or local network may abuse the system to probe or intercept the information such as hosts on the local network, services on the hosts, known vulnerabilities and the packets travelling on the local network in the IT system. |
| R or L | - | Any | T.AUDACC | Persons may not be accountable for the actions that they conduct because the audit records are not reviewed, thus allowing an attacker to escape detection. |

#### 4.3.2.2 Threats Countered by the Operating Environment

Threats countered not by the IDS but the its operational environment are stated in this sub-section. In table 4.4, these threats have the same presentation style with the threats in previous section (section 4.3.2.1).

Table 4.4: Threat to be Countered by Operating Environment

| Initial Privilege Level | Method of Transition | New Level or Action | Threat Name | Threat Description |
|---|---|---|---|---|
| U or S | - | Use or Admin. | T.TUSAGE | The TOE may be inadvertently configured, used and administered in an insecure manner by either authorised or unauthorised persons (DOD, 2000). |

## 4.4 Security Objectives

There are two types of security objectives which need to distinguished in a PP:

- Security objectives for the TOE, which will be satisfied by technical (IT) countermeasures implemented by the TOE;
- Security objectives for the environment, which are to be satisfied by either technical measures implemented by the IT environment, or by non-IT (e.g. procedural) measures.

These two types of objectives for IDS and PcLab LAN environment are presented in this sub-section. While writing those objectives, each threat, defined in section 4.3, is tried to be countered in full or in part by at least one security objective.

For simplification of referencing the objectives, they are uniquely labelled by "O.", indicating that it is an objective, followed by the selected abbreviation of individual objective (e.g. O.IDAUTH).

### 4.4.1 Information Technology Security Objectives

The following are the IT security objectives for the TOE:

**O.IDAUTH** The TOE must uniquely identify and authenticate the claimed identity of all users, before granting access to TOE functions or, for certain specified services (DOD, 2000).

**O.SINUSE** The TOE must prevent the reuse of authentication data for users attempting to authenticate at the TOE (DOD, 2000).

**O.SELPRO** The TOE must protect itself against attempts by unauthorised users to bypass, deactivate, access or tamper with TOE security functions and data (Stoneburner, 2000; DOD, 2000).

**O.AUDREC** The TOE must provide a means to record a readable audit trail of security-related events, with accurate dates and times, and a means to search and sort the audit trail based on relevant attributes.

**O.ACCOUN** The TOE must provide user accountability for authorised administrator use of security functions related to audit.

**O.SECFUN** The TOE must provide functionality that enables an authorised administrator to use the TOE security functions, and must ensure that

only authorised administrators are able to access such functionality (Common Criteria Org. 2000).

**O.EXPORT** When any IDS component makes its data available to another IDS components, the TOE will ensure the confidentiality of that data (SAIC, 2000).

**O.OFLOWS** The TOE must appropriately handle potential audit and its data storage overflows (SAIC, 2000).

**O.IDSCOL** The collector must collect the network packets, store and supply to analyzer the processed network packets that might be indicative of the potential for a future intrusion or the occurrence of an intrusion of an IT System.

**O.IDSANL** The analyzer must get data from IDS collector(s) and the database; then apply analytical processes and information to derive conclusions about intrusions (past, present, or future) of an IT system.

**O.IDSRESP** The Analyzer should respond appropriately and audit when intrusion is discovered. Depending on the policy, Management Server reacts to the intrusion when the Analyzer sends an intrusion alarm.

### 4.4.2 Security Objectives for the Environment

All of the assumptions stated in Section 4.3.1, except A.ATTACK, are considered to be security objectives for the environment. The following are the Protection Profile non-IT security objectives, which, in addition to those assumptions, are to be satisfied without imposing technical requirements on the TOE. That is, they will not require the implementation of functions in the TOE. Thus, they will be satisfied largely through application of procedural or administrative measures (Department of Defense [DOD], 2000).

**O.PROTECT** The TOE is physically protected from unauthorised modification by potentially hostile outsiders.

**O.ADMIN** One or more non-hostile authorised administrators are assigned who are competent to manage the TOE and the security of the information it contains, and who can be trusted not to deliberately

abuse their privileges so as to undermine security and follow all administrator guidance; however, they are capable of error.

**O.NOREMO** Human users can not access the TOE remotely from the internal or external networks.

**O.IDSNET** IDS is maintained as no information can flow in the internal networks without reaching to the IDS also.

**O.GENPUR** There are no general-purpose computing capabilities (e.g., the ability to execute arbitrary code or applications) and storage repository capabilities on the TOE.

**O.PUBLIC** The TOE does not host public data.

**O.TOECONF** The TOE is installed, configured, and managed in accordance with its evaluated configuration.

**O.AUDIT** The underlying system will audit the actions of system users.

**O.ALLINS** All TOE modules are installed on the same computer.

**O.DATABS** The database as an essential part of IDS is supplied by the third party and it should be evaluated separately from the IDS.

**O.GUIDAN** Those responsible for the TOE must ensure that the TOE is delivered, installed, administered, and operated in a manner that maintains security.

**O.ADMTRA** Authorised administrators are trained as to establishment and maintenance of security policies and practices.

For a detailed mapping between threats, assumptions, and the security objectives listed above see section 4.6 Rationale.

### 4.5 Information Technology Security Requirements

IT security requirements include:

- TOE security requirements (TSF), and (optionally)
- Security requirements for the TOE's IT environment (that is, for hardware, software, or firmware external to the TOE and upon which satisfaction of the TOE's security objectives depends).

These requirements are discussed separately in following sub-sections.

### 4.5.1 Target of Evaluation Security Requirements

The CC divides TOE security requirements into two categories (Computer Sciences Corporation, 1999):

- Security functional requirements (SFRs), that is, requirements for security functions such as information flow control, audit, identification and authentication.
- Security assurance requirements (SARs), provide grounds for confidence that the TOE meets its security objectives (for example, configuration management, testing, vulnerability assessment).

These two categories of TOE security requirements for a PP can be constructed by using the following inputs (Common Criteria 1999c):

a) Existing functional or assurance requirements components: The TOE functional or assurance requirements in a PP or ST may be expressed directly, using the components in Part 2 or 3.

b) Existing packages: Part of the TOE security requirements in a PP or ST may have already been expressed in a package that may be used. A set of predefined packages is the EALs defined in Part 3. The TOE assurance requirements in a PP or ST should include an EAL from Part 3.

c) Extended requirements: Additional functional requirements not contained in Part 2 and/or additional assurance requirements not contained in Part 3 may be used in a PP or ST.

The first stage in the SFR selection process from existing CC requirements is, for each security objective for the TOE, to identify the SFRs, which satisfy them. For the conformance of this satisfaction (ISO/IEC JTC, 1999) and (DOD, 2000) is used as the guidance and reference. Once a complete set of SFRs has been established, there then follows an iterative process whereby the complete dependencies between the SFRs should be satisfied. For instance,

> a) The PP may include a security objective requiring the TOE to provide specific responses to the detection of events indicative of an imminent security violation. This leads to the inclusion of a *principal* SFR based on the FAU_ARP.1 (Security Alarms) component.

b) According to CC, FAU_ARP.1 has a dependency on FAU_SAA.1 (Potential Violation Analysis) which should also be included as a *supporting* SFR.

c) FAU_SAA.1 has a dependency on FAU_GEN.1 (Audit Data Generation).

d) FAU_GEN.1 has a dependency on FPT_STM.1 (Reliable Time Stamps).

e) FPT_STM.1 introduces no requirements for additional functional components.

CC permits a degree of flexibility in the way the SFRs and SARs are specified by allowing a set of *operations* to be performed on them to tailor the requirement appropriately. These are (ISO/IEC JTC, 1999):

a) *assignmen*t, where CC allows the inclusion of details specific to a given SFR, e.g. a series of access control rules to be applied, or a list of subjects or objects to which the SFR applies;

b) *iteratio*n, where CC allows a specific functional or assurance component to be used more than once in a PP;

c) *selectio*n, where CC allows one or more specific options to be selected from a given list, e.g. to define whether the SFR applies to the *displaying* and/or *modification* of particular data items;

d) *refinemen*t, where ISO 15408 allows the PP or ST author to add details to a functional or assurance component, for example to restrict the possible solutions to a given SFR or SAR, or to explain in what conditions the refinement is used.

A final point to note is that each security requirement component in CC Part 2 and Part 3 is assigned its own unique reference in CC, based on a defined taxonomy. For example, the component FAU_GEN.1.2 has the following meaning (ISO/IEC JTC, 1999):

a) 'F' indicates it is a *functional* requirement;

b) 'AU' indicates it belongs to the *security audit* class of SFRs;

c) 'GEN' indicates it belongs to the *security audit data generation* family within that class;

d) '1' indicates it belongs to the *audit data generation* component within that family;

e) ‘2’ indicates it is the second *element* within that component.

However, in following section labels of requirements are written in open format for the first time of their statement in order to increase readability.

#### 4.5.1.1 Target of Evaluation Security Functional Requirements

This section presents the SFRs for the TOE.

**Functional security AUdit audit data GENeneration.1: Audit data generation**

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) All auditable events for the not specified level of audit; and

c) [the events listed in Table 4.5].

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [information specified in column three of Table 4.5].

Table 4.5: Auditable Events (cont. on next page).

| Functional Component | Auditable Event Contents | Additional Audit Record |
|---|---|---|
| FAU_ARP.1, FID_ARP.1 | Actions taken due to imminent security violations. | |
| FID_DCL1 | Network packet data from TOE network interface(s) | ID, protocol, source address, destination address, source port, destination port, data portion of the packet |
| FAU_SAA.1, 2.3.4 FID_IDA.1,2,3 | Intrusion data | ID, intrusion type, priority, event ID, module ID |
| FAU_SAR.1 | Reading of information from the audit records | |
| FAU_SEL.1 | All modifications to the audit configuration that occur while the audit collection functions are operating | |

Table 4.5: Auditable Events (cont.).

| Functional Component | Auditable Event Contents | Additional Audit Record |
|---|---|---|
| FMT_SMR.1 | Modifications to the group of users that are part of the authorized administrator role | The identity of the authorized administrator performing the modification and the user identity being associated with the authorized administrator role |
| FIA_UID.2 | All use of the user identification mechanism. | The user identities provided to the TOE |
| FIA_UAU.1 | All use of the authentication mechanism. | The user identities provided to the TOE |
| FIA_AFL.1 | The reaching of the threshold for unsuccessful authentication attempts and the subsequent restoration by the authorized administrator of the users capability to authenticate. | The identity of the offending user and the authorized administrator |
| FPT_STM.1 | Changes to the time | The identity of the authorized administrator performing the operation |
| FMT_MOF.1 | Use of the functions listed in this requirement pertaining to audit. | The identity of the authorized administrator performing the operation |

**FAU Security Audit Review.1: Audit review**

FAU_SAR.1.1 The TSF shall provide [an authorized administrator] with the capability to read [all audit trail data] from the audit records.

FAU_SAR.1.2 The TSF shall provide the audit records in a manner suitable for the user to interpret the information.

**FAU SAR.3: Selectable Audit Review**

FAU_SAR.3.1 The TSF shall provide the ability to perform searches, sorting, ordering of audit data based on [all the available attributes of audit data with logical relations].

**FAU security audit event SELection.1: Selective audit**

FAU_SEL.1.1 The TSF shall be able to include or exclude auditable events from the set of audited events based on the following attributes:

a) event type

b) [and the additional attributes that the event has].

**FAU security audit event SToraGe.1: Protected audit trail storage**

FAU_STG.1.1 The TSF shall protect the stored audit records from unauthorized deletion.

FAU_STG.1.2 The TSF shall be able to prevent modifications to the audit records.

**FAU STG.4: Prevention of audit data loss**

FAU_STG.4.1 The TSF shall prevent auditable events except those taken by the authorized administrator, [shall limit the number of audit records lost and send an alarm to management server] if the audit trail is full.

**FAU security audit Automatic ResPonse.1: Security alarms**

FAU_ARP.1.1 The TSF shall take [one or more of following disruptive actions as the response] upon detection of a potential security violation.

Assignment: Alerts graded according to their seriousness having capability to be activated via different notification channels to different alert destinations, such as e-mail to administrator, alert lines on user-interface and reports; closure of session; resetting the connection; deny, degrade, disturb and deceiving the intruder; and reconfiguring computer systems.

**FAU Security Audit Analysis.1: Potential violation analysis**

FAU_SAA.1.1 The TSF shall be able to apply a set of rules in monitoring the audited events and based upon these rules indicate a potential violation of the TSP.

FAU_SAA.1.2 The TSF shall enforce the following rules for monitoring audited events:

a) Accumulation or combination of [audited network packet data] known to indicate a potential security violation;

b) [assignment: any other rules].

**FAU SAA.2: Profile based anomaly detection**

FAU_SAA.2.1 The TSF shall be able to maintain profiles of system usage, where an individual profile represents the historical patterns of usage performed by the member(s) of [network groups such as insider users or outsiders].

FAU_SAA.2.2 The TSF shall be able to maintain a suspicion rating associated with each user whose activity is recorded in a profile, where the suspicion rating represents the degree to which the user's current activity is found inconsistent with the established patterns of usage represented in the profile.

FAU_SAA.2.3 The TSF shall be able to indicate an imminent violation of the TSP when a user's suspicion rating exceeds the following threshold conditions [which are settable automatically by the TOE or manually by the administrator].

**FAU SAA.3: Simple attack heuristics**

FAU_SAA.3.1 The TSF shall be able to maintain an internal representation of the following signature events [generated signatures from network packet data] that may indicate a violation of the TSP.

FAU_SAA.3.2 The TSF shall be able to compare the signature events against the record of system activity discernible from an examination of [network packets data].

FAU_SAA.3.3 The TSF shall be able to indicate an imminent violation of the TSP when a system event is found to match a signature event that indicates a potential violation of the TSP.

**FAU SAA.4: Complex attack heuristics**

FAU_SAA.4.1 The TSF shall be able to maintain an internal representation of the following event sequences of known intrusion scenarios [assignment: list of sequences of system events whose occurrence are representative of known penetration scenarios (Those scenarios are very dynamic and product specific that's why they are not listed here.)] and the following signature events [based on the network

packet data (Those signatures are very dynamic and product specific that's why they are not listed here.)] that may indicate a potential violation of the TSP.

FAU_SAA.4.2 The TSF shall be able to compare the signature events and event sequences against the record of system activity discernible from an examination of [audited network packet data].

FAU_SAA.4.3 The TSF shall be able to indicate an imminent violation of the TSP when system activity is found to match a signature event or event sequence that indicates a potential violation of the TSP.

**Functional Identification & Authentication Authentication FaiLure.1: Authentication failure handling**

FIA_AFL.1.1 The TSF shall detect when [a non-zero number settable by {an authorized administrator}] of unsuccessful authentication attempts occur related to [users not associated with the authorized administrator role attempting to authenticate from an internal or external network].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall [prevent the offending user from successfully authenticating until the administrator takes some action to make authentication possible for the user in question].

**FIA user ATtribute Definition.1: User attribute definition**

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users:

a) [Identity

b) association of a human user with the authorized administrator role;

c) {no other user security attributes}].

**FIA User AUthentication.1: Timing of authentication**

FIA_UAU.1.1 The TSF shall allow [no action] on behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

**FIA_UAU.4: Single-use authentication mechanisms**

FIA_UAU.4.1 The TSF shall prevent reuse of authentication data related to [assignment: identified authentication mechanism(s)].

**FIA User IDentification.2: User identification before any action**

FIA_UID.2.1 The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on behalf of that user.

**Functional security ManagmenT Managment Of Functions in TSF.1: Management of security functions behavior**

FMT_MOF.1.1 The TSF shall restrict the ability to [perform] the functions:

a) [start up and shut down of the TOE,

b) adding, modifying, deletion of rules, system events and system event sequences (FID_IDA),

c) enable, disable, determine and modify the behaviour of all other TOE functions to [the authorized administrator].

**FMT Managment Of TSF Data.1: Management of TSF data**

FMT_MTD.1.1 The TSF shall restrict the ability to change default, query, modify, delete, change, assign and add [all TOE data and attributes] to [the authorized administrator].

**FMT_MTD.2: Management of limits on TSF data**

FMT_MTD.2.1 The TSF shall restrict the specification of the limits for [the number of authentication failures] to [the authorized administrator].

FMT_MTD.2.2 The TSF shall take the following actions, if the TSF data are at, or exceed, the indicated limits: [actions specified in FIA_AFL.1.2].

**FMT Security Management Role.1: Security roles**

FMT_SMR.1.1 The TSF shall maintain the role [authorized administrator].

FMT_SMR.1.2 The TSF shall be able to associate human users with the authorized administrator role.

**Functional Protection of the TSF Internal TOE TSF data Transfer.1: Basic internal TSF data transfer protection**

FPT_ITT.1.1 The TSF shall protect TSF data from both disclosure and modification when it is transmitted between separate parts of the TOE.

**FPT Refernece V Mediation.1: Non-bypassability of the TSP**

FPT_RVM.1.1 The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

**FPT domain SEParation.1: TSF domain separation**

FPT_SEP.1.1 The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.1.2 The TSF shall enforce separation between the security domains of subjects in the TSC.

**FPT time STaMps.1: Reliable time stamps**

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps for its own use.

**Proposed Intrusion Detection Class Requirements**

CC recognises that there may be cases where there is no appropriate functional or assurance component in CC Part 2 or CC Part 3. In this case, the IT security requirements may be stated explicitly without reference to CC; however, such requirements must be unambiguous, evaluatable, and expressed in a similar style to existing CC components. On this ground of allowance, a Proposed Intrusion Detection Class is written and presented at Appendix D. The reason of writing a new requirement

class is that there is no any class in CC Part 2 for defining real-time intrusion detection requirements. This new class, which has similar style of Functional Security Audit Class (FAU), includes some components related to collection of data, analyze of collected data and response of detected intrusions.

**Functional Intrusion Detection Automatic ResPonse.1: Security alarms**

FID_ARP.1.1 The TSF shall take [one or more of following disruptive actions as the response] upon detection of an intrusion.

Assignment: Alerts graded according to their seriousness having capability to be activated via different notification channels to different alert destinations, such as e-mail to administrator, alert lines on user-interface and reports; closure of session; resetting the connection; deny, degrade, disturb and deceiving the intruder; and reconfiguring computer systems.

**FID Data CoLlection.1: Data Collection**

FID_DCL.1.1 The TSF shall be able to collect, [store in the database and/or sent for the analysis] [network packet data from TOE network interface(s)] from the IT system resources.

**FID Intrusion Detection Analysis.1: Intrusion detection analysis**

FID_IDA.1.1 The TSF shall be able to apply a set of rules in monitoring the collected event data and based upon these rules indicate an intrusion.

FID_IDA.1.2 The TSF shall enforce the following rules for monitoring collected data:

a) Accumulation or combination of [assignment: *subset of defined collectable event data*] known to indicate an intrusion;

b) [assignment: *any other rules*].

**FID_IDA.2: Simple attack heuristics**

FID_IDA.2.1 The TSF shall be able to maintain an internal representation of the

following signature events [signatures generated from network packets data] that may indicate an intrusion.

FID_IDA.2.2 The TSF shall be able to compare the signature events against the collected system data discernible from an examination of [network packets data].

FID_IDA.2.3 The TSF shall be able to indicate an imminent intrusion when a system event is found to match a signature event that indicates an intrusion.

**FID_IDA.3: Complex attack heuristics**

FID_IDA.3.1 The TSF shall be able to maintain an internal representation of the following event sequences of known intrusion scenarios [assignment: *list of sequences of system events whose occurrence are representative of known penetration scenarios* (Those scenarios are very dynamic and product specific that's why they are not listed here.)] and the following signature events [based on the network packet data (Those signatures are very dynamic and product specific that's why they are not listed here.)] that may indicate an intrusion.

FID_IDA.3.2 The TSF shall be able to compare the signature events and event sequences against the collected system data discernible from an examination of [network packet data].

FID_IDA.3.3 The TSF shall be able to indicate an imminent intrusion when system activity is found to match a signature event or event sequence that indicates an intrusion.

#### 4.5.1.2 TOE Security Assurance Requirements

Table 4.6 identifies the security assurance components drawn from CC Part 3: Security Assurance Requirements; these requirements are restated in Appendix C.

Table 4.6: Restated TOE Security Assurance Requirements (cont. on next page).

| Class | Assurance Component ID | Assurance Component Name |
|---|---|---|
| ACM: Configuration management | ACM_CAP.2 | Configuration items |
| ADO Delivery and operation | ADO_DEL.1 | Delivery procedures |
| | ADO_IGS.1 | Installation, generation, and start-up procedures |

Table 4.6: Restated TOE Security Assurance Requirements (cont.).

| Class | Assurance Component ID | Assurance Component Name |
|---|---|---|
| ADV: Development | ADV_FSP.1 | Informal functional specification |
| | ADV_HLD.1 | Descriptive high- level design |
| | ADV_RCR.1 | Informal correspondence demonstration |
| AGD: Guidance documents | AGD_ADM.1 | Administrator guidance |
| | AGD_USR.1 | User guidance |
| ATE: Tests | ATE_COV.1 | Evidence of coverage |
| | ATE_FUN.1 | Functional testing |
| | ATE_IND.2 | Independent testing - sample |
| AVA: Vulnerability assessment | AVA_SOF.1 | Strength of TOE security function evaluation |
| | AVA_VLA.1 | Developer vulnerability analysis |

### 4.5.2 Security Requirements for the Information Technology Environment

The TOE has no security requirements allocated to its IT environment.

## 4.6 Rationale

This section demonstrates the completeness and consistency of this PP. The security objectives for the IT and environment are explained in terms of threats countered and assumptions met. The security functional requirements (SFR) are explained in terms of objectives met by the requirement. The traceability and consistency is illustrated through matrices that map the following:

- security objectives to threats countered
- objectives to assumptions met
- SFRs to objectives met

### 4.6.1 Rationale For Information Technology Security Objectives

In Table 4.7, mapping between objectives and threats are presented. By this way, it would be possible to see which threat is countered by which objective. One threat must be countered by at least one objective, on the other hand many objectives can counter the same threat. These objectives should be satisfied by IDS requirements.

Table 4.7: Summary of Mappings Between Threats, Policies and IT Security Objectives.

| | T.NOAUTH | T.REPEAT | T.REPLAY | T.SELPRO | T.DISALT | T.AUDFUL | T.AUDACC | T.NOHALT | T.AUSEVIR | T.AUSEDENY | T.BUGPRIV | T.MISINT | T.AUSEINT | T.TUSAGE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **O.IDSCOL** | | | | | X | | | | | X | X | X | X | |
| **O.IDSANL** | | | | | X | | | | | X | X | X | X | |
| **O.IDSRESP** | | | | | X | | | X | | X | X | X | X | |
| **O.IDAUTH** | X | | | | X | | | X | | | | | | |
| **O.SINUSE** | | X | X | | | | | | | | | | | |
| **O.SELPRO** | | X | | X | X | X | | X | | | | | | |
| **O.AUDREC** | | | | | | | X | | | | | | | |
| **O.ACCOUN** | | | | | | | X | | | | | | | |
| **O.SECFUN** | X | | X | | | X | | | | | | | | |
| **O.EXPORT** | | | | | X | | | | | | | | X | |
| **O.OFLOWS** | | | | | | X | | X | | | | | | |

### 4.6.2 Rationale For Security Objectives for the Environment

Table 4.8 represents mapping and tracibility between security objectives which are satisfied by the working environment and threats.

Table 4.8: Summary of Mapping Between Threats and Security Objectives for the Environment.

| | T.NOAUTH | T.REPEAT | T.REPLAY | T.SELPRO | T.DISALT | T.AUDFUL | T.AUDACC | T.NOHALT | T.AUSEVIR | T.AUSEDENY | T.BUGPRIV | T.MISINT | T.AUSEINT | T.TUSAGE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **O.GUIDAN** | | | | | | | X | | | | | | | X |
| **O.ADMTRA** | | | | | | | X | | | | | | | X |

Since the rest of the security objectives for the environment are, in part 4.4.2, statement of the security assumptions, those security objectives trace to all aspects of the assumptions. Thus they are not quoted in table 4.8.

### 4.6.3 Rationale For Security Requirements

As being the last section of Rationale, mapping between security functional requirements and objectives is presented in Table 4.9.

Table 4.9: Summary of Mappings Between Functional Requirements and Objectives for the TOE (cont. on next page).

| | O.IDSCOL | O.IDSANL | O.IDSRESP | O.IDAUTH | O.SINUSE | O.SELPRO | O.AUDREC | O.ACCOUN | O.SECFUN | O.EXPORT | O.OFLOWS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FAU_ARP.1 | | | X | | | | | | | | |
| FAU_GEN.1 | X | X | | | | | X | X | | | |
| FAU_.SAA.1 | | X | | | | | | | | | |
| FAU_.SAA.2 | | X | | | | | | | | | |
| FAU_.SAA.3 | | X | | | | | | | | | |
| FAU_.SAA.4 | | X | | | | | | | | | |
| FAU_SAR.1 | | | | | | | X | | | | |
| FAU_SAR.3 | | | | | | | X | | | | |
| FAU_STG.1 | | | | | | X | | | X | | X |
| FAU_STG.4 | | | | | | X | | | X | | X |
| FIA_AFL.1 | | | | | | X | | | | | |
| FIA_ATD.1 | | | | X | X | | | | | | |
| FIA_UAU.1 | | | | X | X | | | | | | |
| FIA_UAU.4 | | | | | X | | | | | | |
| FIA_UID.2 | | | | X | | | | X | | | |
| FMT_MOF.1 | | | | | | | | X | X | | |
| FMT_MTD.1 | | | | | | | | | X | | |
| FMT_MTD.2 | | | | | | | | | X | | |
| FMT_SMR.1 | | | | | | | | | X | | |
| FPT_ITT.1 | | | | | | | | | | X | |
| FPT_RVM.1 | | | | | | X | | | | | |
| FPT_SEP.1 | | | | | | X | | | | | |
| FPT_STM.1 | | | | | | | X | | | | |
| FID_ARP.1 | | | X | | | | | | | | |

Table 4.9: Summary of Mappings Between Functional Requirements and Objectives for the TOE (cont.).

| | O.IDSCOL | O.IDSANL | O.IDSRESP | O.IDAUTH | O.SINUSE | O.SELPRO | O.AUDREC | O.ACCOUN | O.SECFUN | O.EXPORT | O.OFLOWS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FID_DCL.1 | X | | | | | | | | | | |
| FID_IDA.1,2,3 | | X | | | | | | | | | |

# Chapter 5

# INTRUSION DETECTION SYSTEM PROTECTION PROFILE IMPLEMENTATION ON PRODUCT EVALUATION

Internet Security Systems (ISS) RealSecure™ (Version 6.0) is evaluated against the Intrusion Detection System Protection Profile for PC Laboratory LAN Environment (Chapter 4) in sub-section 5.2. A brief overview about RealSecure IDS is presented before this evaluation in sub-section 5.1.

## 5.1 RealSecure Version 6.0

Realsecure is an automated real-time intrusion detection system that uses built-in attack signatures and statistical profiles to detect different types of misuse for computer networks and hosts. It consists of following basic sub-systems:

- RealSecure Network Sensor runs on a dedicated system that monitors network traffic for attack signatures – definitive identifiers that an intrusion is underway. Attack recognition, incident response, and intrusion prevention occur immediately, with customization of signatures and response capabilities.
- OS Sensor is installed on hosts in the network and monitors the host audit logs and other host-specific data sources.
- RealSecure Server Sensor performs real-time intrusion monitoring, detection, and prevention of malicious activity by analyzing kernel-level events, host logs, and network activity on critical servers. Server sensor monitors, detects, and prevents intrusions with packet interception, blocking capability, and automated correlation analysis using security fusion technology.
- RealSecure Workgroup Manager provides centralized, scalable management, configuration, reporting, and real-time alarming for all RealSecure Sensors. The Workgroup Manager is based on a three-tiered architecture, and supports an enterprise database, MSDE or Microsoft SQL (Internet Security Systems, 2001a; Internet Security Systems, 2001b). The management console displays real-time alarm data in a standard Windows activity tree mode, where the data in the tree can be sorted by destination address, source address, or event name. Events contain an

icon that indicates the severity as well as a distinct event name. Multiple occurrences of the same event are combined into a single notification. Event data (detected potential intrusion event. “Event” is used with same meaning of “intrusion event” in RealSecure literature.) can also be stored in an ODBC-compliant database for generation of reports. Reports are available in text and graphic formats and the user can launch customized reports from the interface. Reporting features allow the administrator to sort and format event data by priority, source address, destination address, or network service over some period of time (Jackson, 1999).

RealSecure sensors use policies that control what the sensor monitors and how it responds to events that it detects. The features of the network sensor are controlled through these policies which can be either the pre-defined policies that ship with the sensors, or customized policies. Following operations are allowed for customizing the policies:

- customize (fine-tune) pre-defined signatures where a signature is the internal code that RealSecure uses to detect an event, or series of events, that might signal an attack on network or that can provide security-related information.
- determine how the sensor responds to events where responds include recording the date, time, source, and target of the event; recording the content of the event; notifying the network administrator; reconfiguring the firewall; terminating the event automatically.
- monitor attempted and successful network connections.
- have the sensor ignore specific traffic to and from trusted computers
- create custom signatures to monitor specific activity that other signatures do not detect.

### 5.2 Evaluation of RealSecure Version 6.0

According to CC, evaluations are carried out by accredited evaluation facilities (testing laboratories). The evaluation process is illustrated in Figure 5.1. This procedure includes three basic steps (Common Criteria Org. 1999a):

1. **PP evaluation:** The PP evaluation is carried out against the evaluation criteria for PPs contained in Part 3 (Common Criteria Org. 1999c). The goal of such an

evaluation is to demonstrate that the PP is complete, consistent, and technically sound and suitable for use as a statement of requirements for an evaluatable TOE.

2. **ST evaluation:** The evaluation of the ST for the TOE is carried out against the evaluation criteria for STs contained in Part 3 (Common Criteria Org. 1999c). The goal of such an evaluation is twofold: first to demonstrate that the ST is complete, consistent, and technically sound and hence suitable for use as the basis for the corresponding TOE evaluation; second, in the case where an ST claims conformance to a PP, to demonstrate that the ST properly meets the requirements of the PP.

3. **TOE evaluation:** The TOE evaluation is carried out against the evaluation criteria contained in Part 3 (Common Criteria Org. 1999c) using an evaluated ST as the basis. The goal of such an evaluation is to demonstrate that the TOE meets the security requirements contained in the ST.

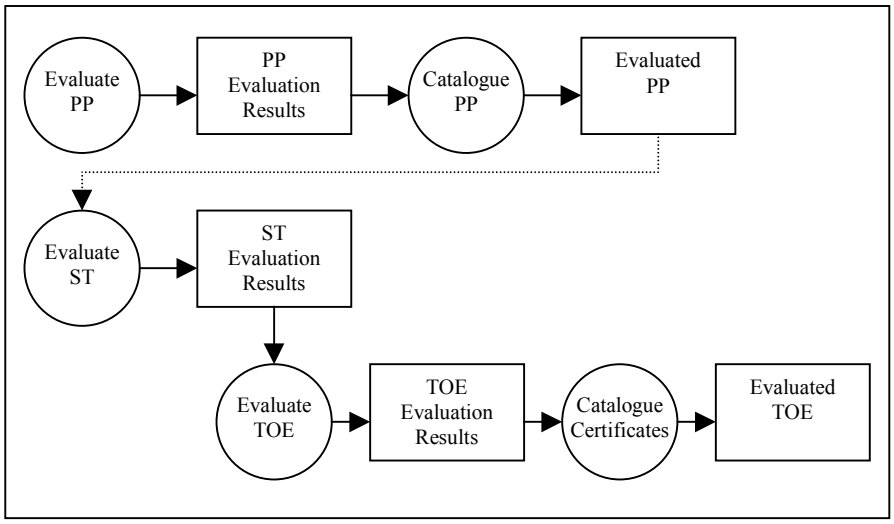


Figure 5.1: Evaluation Process and Results (Common Criteria Org. 1999a).

The IDS PP for PcLab LAN Environment is used here directly for evaluation of RealSecure without any evaluation of testing laboratories and ST. For the evaluation of RealSecure, Table 5.1 (functional requirements) and Table 5.2 (assurance requirements) are created based on IDS PP for PcLab LAN Environment, presented in Chapter 4. The first three columns of these tables identify the assessed requirement with its ID, name and element number. The individual requirement assessments at each row may get the following four result values:

- Not satisfied (N); Requirement is not included in product.

- Partially satisfied (P); Product includes specification(s) which meet(s) the requirement but does/do not exactly fit the definition of requirement.
- Satisfied (S); Product includes specification(s) which exactly fit(s) the requirement definition.
- Unknown (U): This requirement could not be tested with available information sources.

These results are represented at "Result" column of the tables and the additional descriptions are done at "description" column if needed. (The details of evaluated requirements can be found at Chapter 4 and Appendix C.) Following three main sources are used as the bases of this evaluation:

- Documentation of RealSecure (Internet Security Systems, 2001a, 2001b, 2001c, 2001d, 2001e).
- RealSecure v6.0 installed on İzmir Institute of Technology, Department of Computer Engineering, PC Laboratory. This installation, used in the evaluation, includes network sensor and management console only.
- Previous reviews and assesments on RealSecure such as Jackson, 1999 and ICSA Labs, 1999.

Table 5.1: ISS RealSecure Functional Requirements Assessment Table (cont. on next page).

| **Functional Component ID** | **Functional Component Name** | **Element No** | **Result** | **Description** |
|---|---|---|---|---|
| FAU Security Audit Automatic Response | | | | _____ |
| FAU_ARP.1 | Security alarms | 1 | N | Security alarms are not resulted from audit data analysis. |
| FAU Security Audit Generation | | | | _____ |
| FAU_GEN.1 | Audit data generation | 1 | P | Only detected potential intrusion event data is audited. |
| | | 2 | P | Audit data details are satisfied only for intrusion event data. |
| FAU Security Audit Analysis | | | N | This evaluated installation does not perform any audit data analysis because it includes only network sensor. |

Table 5.1: ISS RealSecure Functional Requirements Assessment Table (cont.).

| Functional Component ID | Functional Component Name | Element No | Result | Description |
|---|---|---|---|---|
| FAU_SAA.1 | Potential violation analysis | 1 | N | _____ |
| | | 2 | N | _____ |
| FAU_SAA.2 | Profile based anomaly detection | 1 | N | _____ |
| | | 2 | N | _____ |
| | | 3 | N | _____ |
| FAU_SAA.3 | Simple attack heuristics | 1 | N | _____ |
| | | 2 | N | _____ |
| | | 3 | N | _____ |
| FAU_SAA.4 | Complex attack heuristics | 1 | N | _____ |
| | | 2 | N | _____ |
| | | 3 | N | _____ |
| FAU Security Audit Review | | | | _____ |
| FAU_SAR.1 | Audit review | 1 | P | Not only the administrator but also all the o/s users can view audit records, which are stored in database. |
| | | 1 | S | _____ |
| FAU_SAR.3 | Selectable audit review | 1 | P | Only sorting and ordering operations available on audit records. |
| FAU Security Audit Event Selection | | | | _____ |
| FAU_SEL.1 | Selective audit | 1 | S | _____ |
| FAU Security Audit Event Storage | | | | _____ |
| FAU_STG.1 | Protected audit trail storage | 1 | N | _____ |
| | | 2 | N | _____ |
| FAU_STG.4 | Prevention of audit data loss | 1 | N | _____ |
| FIA Authentication Failure | | | N | There is no any authentication mechanism for the users. |
| FIA_AFL.1 | Authentication failure handling | 1 | N | _____ |
| | | 2 | N | _____ |
| FIA User Attribute Definition | | | N | There is no any user attribute definition. |
| FIA_ATD.1 | User attribute definition | 1 | N | _____ |
| FIA User Authentication | | | N | There is no any authentication mechanism for the users. |
| FIA_UAU.1 | Timing of authentication | 1 | N | _____ |
| | | 2 | N | _____ |
| FIA_UAU.4 | Single-use authentication mechanisms | 1 | N | _____ |

Table 5.1: ISS RealSecure Functional Requirements Assessment Table (cont.).

| Functional Component ID | Functional Component Name | Element No | Result | Description |
|---|---|---|---|---|
| FIA User Identification | | | N | There is no any identification mechanism. |
| FIA_UID.2 | User identification before any action | 1 | N | _____ |
| FMT Managment of Functions | | | | _____ |
| FMT_MOF.1 | Management of security functions behavior | 1 | P | Work Group Manager (Management Module) based restrictions are available. |
| FMT Management of Data | | | N | _____ |
| FMT_MTD.1 | Management of TSF data | 1 | N | _____ |
| FMT_MTD.2 | Management of limits on TSF data | 1 | N | _____ |
| | | 2 | N | _____ |
| FMT Security Management Role | | | N | _____ |
| FMT_SMR.1 | Security roles | 1 | N | _____ |
| | | 2 | N | _____ |
| FPT Internal Data Transfer | | | | _____ |
| FPT_ITT.1 | Basic internal TSF data transfer protection | 1 | P | Only encryption is available. |
| FPT Reference Mediation | | | | _____ |
| FPT_RVM.1 | Non-bypassability of the TSP | | U | _____ |
| FPT Domain Separation | | | | _____ |
| FPT_SEP.1 | TSF domain separation | 1 | U | _____ |
| | | 2 | U | _____ |
| FPT Time Stamps | | | | _____ |
| FPT_STM.1 | Reliable time stamps | 1 | U | _____ |
| FID Automatic Response | | | | _____ |
| FID_ARP.1 | Security alarms | 1 | S | Real-time alarms via console and different ways. |
| FID Data Collection | | | | |
| FID_DCL.1 | Data Collection | 1 | P | Only collects but does not store collected data. |
| FID Intrusion Detection Analysis | | | | _____ |
| FID_IDA.1 | Intrusion detection analysis | 1 | S | _____ |
| | | 2 | S | _____ |

Table 5.1: ISS RealSecure Functional Requirements Assessment Table (cont.).

| Functional Component ID | Functional Component Name | Element No | Result | Description |
|---|---|---|---|---|
| FID_IDA.2 | Simple attack heuristics | 1 | S | _____ |
| | | 2 | S | _____ |
| | | 3 | S | _____ |
| FID_IDA.3 | Complex attack heuristics | 1 | U | _____ |
| | | 2 | U | _____ |
| | | 3 | U | _____ |

The Table 5.2 has the similar structure with Table 5.1. Additionally, in the third column of Table 5.2, "C and D" are used to indicate the Developer action elements, Content and presentation of elements respectively. The evaluator action elements of assurance requirements are not included in Table 5.2 and not evaluated, since RealSecure has not been evaluated by any evaluation labratory (see Appendix C for details of those three sets of element).

Table 5.2: ISS RealSecure Security Assurance Requirements Assessment Table (cont. on next page).

| Assurance Component ID | Assurance Component Name | Element No | Result | Description |
|---|---|---|---|---|
| ACM_CAP.2 | Configuration items | | | _____ |
| | | 1 C, D | S | _____ |
| | | 2 C, D | S | _____ |
| | | 3 C, D | S | Included in user-guides, not a separated configuration document |
| | | 4 C | S | _____ |
| | | 5 C | S | _____ |
| | | 6 C | S | _____ |
| ADO_DEL.1 | Delivery procedures | | | _____ |
| | | 1 C, D | U | Not able to reach or find out any delivery procedure. |
| | | 2 D | U | _____ |
| ADO_IGS.1 | Installation, generation, and start-up procedures | 1 C, D | S | Developer supplies installation guides for each module. |

Table 5.2: ISS RealSecure Security Assurance Requirements Assessment Table (cont.).

| Assurance Component ID | Assurance Component Name | Element No | Result | Description |
|---|---|---|---|---|
| ADV_FSP.1 | Informal functional specification | | | _____ |
| | | 1 C, D | S | _____ |
| | | 2 C | U | Internal consistency does not evaluated. |
| | | 3 C | P | Error messages and exceptions are not included in functional specifications. |
| | | 4 C | S | _____ |
| ADV_HLD.1 | Descriptive high- level design | | | _____ |
| | | 1 C, D | S | _____ |
| | | 2 C | U | Internal consistency does not evaluated and unknown. |
| | | 3 C | S | _____ |
| | | 4 C | S | _____ |
| | | 5 C | S | _____ |
| | | 6 C | S | _____ |
| | | 7 C | S | _____ |
| ADV_RCR.1 | Informal correspondence demonstration | 1 C, D | U | Such a demonstration can not be found in public domain. |
| AGD_ADM.1 | Administrator guidance | | | There are no different roles in RealSecure such as user and administrator so user and administrator guidances are the same. |
| | | 1 C, D | S | _____ |
| | | 2 C | S | _____ |
| | | 3 C | S | _____ |
| | | 4 C | S | _____ |
| | | 5 C | S | _____ |
| | | 6 C | S | _____ |
| | | 7 C | U | _____ |
| | | 8 C | P | Guidence is describing some security requirements for the IT environment but not the “all”. |
| AGD_USR.1 | User guidance | | P | There are no different roles in RealSecure such as user and administrator so user and administrator guidances are the same. Hoverer, RealSecure guidences have similar content. |
| ATE_COV.1 | Evidence of coverage | C, D | N | _____ |
| ATE_FUN.1 | Functional testing | | | _____ |
| | | 1 C, D | N | _____ |
| | | 2 C, D | N | _____ |
| | | 3 C | N | _____ |
| | | 4 C | N | _____ |
| | | 5 C | N | _____ |

Table 5.2: ISS RealSecure Security Assurance Requirements Assessment Table(cont.).

| Assurance Component ID | Assurance Component Name | Element No | Result | Description |
|---|---|---|---|---|
| ATE_IND.2 | Independent testing - sample | 1 C, D | U | _____ |
| | | 2 C | U | _____ |
| AVA_SOF.1 | Strength of TOE security function evaluation | 1 C, D | U | _____ |
| | | 2 C | U | _____ |
| AVA_VLA.1 | Developer vulnerability analysis | 1 C | N | _____ |
| | | 2 D | N | _____ |

The following conclusions are drawn from the evaluation of RealSecure requirement assessment:

- Intrusion Detection: Applied intrusion detection mechanisms are not clearly identified in detail so that RealSecure is a "black-box" for its intrusion detection techniques. It fits basic collect, analyze, detect and respond IDS approach in real-time and implements it on client-server architecture. Its documentation (Internet Security Systems, 2001e) says that RealSecure applies signature based intrusion detection technique somehow, however the details of this applied technique are not explained.

- Auditing: RealSecure does not audit the "important events" related to its own operations such as errors, start up and shutdown of its functionalities and so on. It audits detected potential intrusions, this intrusions can be viewed on the GUI or in report format according to their attributes such as priority, intrusion type and so on.

- Authentication and Identification: Although there is no user identification capability for the users, modules have their own identity and password. Management module should use this predefined identity and password of the sensor in order to connect to that sensor. So there is a module based identification and authentication. Since RealSecure does not have any user identification and authentication functionality before allowing any operation, so it should be better to install and run it in the physically separated/isolated and secured environment. Otherwise any valid user of its underlying o/s can manage it and use all functionality provided by RealSecure.

- Management: As RealSecure does not have any user identification, security administrator can not give different roles to different underlying IT system users and different restrictions can not be applied on different roles. Different sensors can be associated by different management consoles and all management jobs can be performed by only those consoles on the sensors. So management restrictions are applicable module-by-module.
- Data Protection: RealSecure supports the communication encryption between its modules but it does not perform integrity checking on the received data.
- Configuration: RealSecure has a configuration management system. The configuration can be performed either via GUI or by editing configuration script files. However the format of this script does not explained in guidances. Configuration guidance only explains the configuration performed via GUI.
- Documentation and Guidance: The developer supplies most of the documentation defined in the assurance requirements including installation, maintenance, configuration, high level design and functional specification for each module of RealSecure. Developer only gives Getting Started Guide and Installation Guide (Internet Security Systems, 2001d, 2001e) with the purchased product as books. The other documentation supplied by the developer is freely available on the Internet.
- Testing: Although there are some independent test results available on the internet, there is no any test result available supplied by the developer (Internet Security Systems) either done by developer or independent evaluator.

Some requirements available in the IDS PP for PcLab LAN Environment are not evaluated since neither RealSecure does not supply necessary information for those requirements evaluation nor there is no evaluation or test results supplied by either developer or an independent test laboratory.

# Chapter 6

# CONCLUSION AND DISCUSSION

The objective of this thesis is to explore the requirements, standards and guidelines to review, evaluate and possibly to develop an IDS. This chapter concludes this research effort. The overview of previous chapters and research effort will be summarized and conclusions are presented in this summary part first. Then following the conclusion, discussion and recommendations for future work are presented.

Chapter 2 begins with a background information about intrusion, intrusion detection systems, client server architecture and security software quality assurance. After this background information is given, the methodology followed in this research is presented. This methodology is based on the assumption that, in order to develop a Common Criteria Protection Profile for a product, a generic model of this product should be developed first.

In Chapter 3, a generic IDS model is developed first in order to achieve the thesis objective. There are some researches presenting only IDS models and there are some others presenting only requirements for the generic IDSs in the literature (Internet Security Systems, 2000, Metcalf and LaPadula, 2000, Lodin, 1998, Puketza et al. 1994). In this thesis, these models and requirements are combined to obtain a generic IDS model, which is a client-server architecture.

This generic IDS model and its requirements provide to see the IDS picture clearly. However, it does not provide evidence/assurance of competence and consistency of IDS requirements. That's why, in Chapter 4, an IDS Protection Profile (PP) is written to encapsulate security functional and assurance requirements for PC Laboratory LAN environment. During the development process of this PP, Common Criteria Protection Profile development methodology is followed. This development process establishes a level of confidence that the security functions of IDS and the assurance measures applied to them are meet. Common Criteria (CC) PP development methodology includes the following steps: At first step, the IDS, its general IT features and architecture are identified. Studies performed in Chapter 3, generic IDS model and its requirements, can construct the base of this step for IDSs. Then, the environment in which IDS would operate is considered in particular identifying the security problems

and challenges that must be addressed. That identification activity is, in essence, a risk analysis and leads to a statement of general needs or security objectives to be met both by the IDS and its environment.

Part 2 catalogue of CC, includes well-defined and understood security functional requirements that can be used as a standard way of expressing the security requirements for IT products and systems. Security objectives are transformed through the use of that Part 2 catalogue of CC into a set of statements that are coherent and mutually supportive security functional requirements. In addition to the available security requirements in Part 2 of CC, proposed functional requirements are written specific to IDSs.

There are some officially evaluated and registered security product PPs such as Traffic-Filter Firewall Protection Profile For Medium Robustness Environments Version 1.4. Since selection operation of the functional requirements from the Part 2 catalogue is not clearly defined in CC, Traffic Filter Firewall PP is used as a guide, model and example for functional requirement selection.

Finally, based on the desired level of confidence in the security of products to be built, an Evaluation Assurance Level (EAL) from Part 3 is assigned and assurance requirements are restated for this EAL.

It is desirable to submit a PP to an independent testing laboratory for evaluation, to ensure that it is correct, complete, and internally consistent. The evaluated PP may then be entered into a central registry for use by the community to communicate the product security needs to manufacturers, either informally or by incorporation into procurement documents. However, this evaluation necessitates a sponsoring organization covering the evaluation cost. That's why the IDS PP in Chapter 4 has not been evaluated by an independent laboratory.

The developed IDS PP is implemented on evaluation of RealSecure IDS, which is installed on İzmir Instituted of Technology, Department of Computer Engineering, PC Laboratory. This evaluated installation of RealSecure is including one network sensor and management console. Two different tables for security functional and assurance requirements are constructed, based on the IDS PP for PC Lab LAN environment. Then evaluation and conclusions are achieved on these tables. Each defined requirement in IDS PP is evaluated according to its existency in RealSecure.

Requirement assessments are valued by the following logical phrases, satisfied, not satisfied, partially satisfied and unknown.

During the above development and implementation process of IDS PP, the following observation has been made; some of the assurance requirements are at abstract level and not measurable in CC such as “The Target of Evaluation shall be suitable for testing”. Those abstract level requirements make the evaluation comparatively harder.

IDS consumers who are not familiar with IDSs and PP development can use this thesis for simple selection and evaluation of IDS products directly via checklists. The presented requirements can be manipulated into a requirement checklist. Such a simple checklist is introduced in Appendix A as an example. More detailed one can be constructed from the requirements in Chapter 3 according to IDS consumer’s needs.

In this thesis, the CC, requirement presentation format, its development and evaluation methodologies are used and suggested as the standards and guidelines for quality assurance of IDS. In the future, any IDS developer or evaluator, can follow the same methodology with this thesis and use the generic IDS model and IDS PP for PcLab LAN environment. However, definitions of requirements and specifications should be made in more detail and measurable way, for a specific IDS product. For this reason, CC Security Target model can be used for defining product-specific requirement presentation in the further studies.

# REFERENCES


Amoroso,E., Kwapniewski,R. (1998). "A Selection Criteria for Intrusion Detection Systems". *AT&T Laboratories.* 280-288. Retrived June 5, 2001, from http://ieeexplore.ieee.org database.

Anderson,J.P. (1980). "Computer Security Threat Monitoring and Surveillance" Technical Report. James P. Anderson Co. Fort Washington Pennsylvania.

Barber,R. (2001). "The Evolution of Intrusion Detection Systems The Next Step". *Computers & Security, Volume 20, Issue 2,* 132-145.

Common Criteria Org. (1999a). "Common Criteria for Information Technology Security Evaluation Part 1: Introduction and General Model Version 2.1". Retrived October 23, 2001 from www.commoncriteria.org.

Common Criteria Org. (1999b). "Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Requirements Version 2.1". Retrived October 23, 2001 from www.commoncriteria.org.

Common Criteria Org. (1999c). "Common Criteria for Information Technology Security Evaluation Part 3: Security Assurance Requirements Version 2.1". Retrived October 23, 2001 from www.commoncriteria.org.

Common Criteria Org. (1999d). "User Guide". Retrived April 29, 2002 from http://www.commoncriteria.org/supporting_docs/CCUsersGuide.pdf.

Common Criteria Org. (2000). "Database Management System Protection Profile issue 2.1".

Common Criteria Org. (2002a). "Common Criteria Origin". Retrived April 29, 2002 from www.commoncriteria.org/docs/origins.html.

Common Criteria Org. (2002b). "CC An Introduction". Retrived April 29, 2002 from http://www.commoncriteria.org/supporting_docs/CCIntroduction.pdf.

Computer Sciences Corporation. (1999). "Firewall-1 Version 4.0 Security Target V2.4". *Prepared for Check Point Software Technologies Ltd. Prepared by Computer Sciences Corporation.*

Das K.J. (2000). “Attack Development for Intrusion Detection Evaluation” *Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degrees of Bachelor of Science in Computer Science and Engineering and Master of Engineering in Electrical Engineering and Computer Science at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY.*

Department of Defense [DOD]. (2000). "Traffic-Filter Firewall Protection Profile For Medium Robustness Environments Version 1.4". *U.S. Department of Defense.*

Dragon 5.0 Intrusion Detection System. (2002) Enterasys Networks, Inc

Durst,R., Champion,T., Witten,B., Spagnuolo,L. (1999). "Testing and Evaluating Computer IDS". *Communications of The ACM, Vol42,* 53-61.

Heady,R., Maccabe,L.A., Servilla,M. (1990). “The Architecture of a Network Level Intrusion Detection System”. Technical Report, Department of Computer Science, University of New Mexico.

HP Intrusion Detection System 9000. (2001). Hewlett-Packard Company.

ICSA Labs. (1999). "IDS Buyers Guide". *ICSA Labs.* Retrived December 24, 2001, from www.icsalabs.com/html/communities/ids/buyers-guide/guide/index.html.

IEEE Std 730. (1998). “IEEE Standard for Software Quality Assurance Plans”. Institute of Electrical and Electronics Engineers, Inc.

IEEE Std 830. (1998). “IEEE Recommended Practice for Software Requirements SpeciŞcations”. Institute of Electrical and Electronics Engineers, Inc.

IEEE. (1999). IEEE Software Engineering Standards Collection. Institute of Electrical and Electronics Engineers (IEEE) Press.

Information Processing Ltd. (1996a). “An Introduction to Software Testing”. *Information Processing Ltd*. Retrived February 7, 2002, from http://www.itpapers.com database.

Information Processing Ltd. (1996b). “Software Testing and Software Development Lifecycles”. *Information Processing Ltd*. Retrived February 7, 2002, from http://www.itpapers.com database.

Internet Security Systems. (1999) “Intrusion Detection for the Millennium”. ISS Technology Brief.

Internet Security Systems. (2000). "Evaluating an Intrusion Detection Solution-A Strategy for a Successful IDS Evaluation". *Internet Security Systems.* Retrived May 5, 2001, from www.iss.net.

Internet Security Systems. (2001a). “RealSecure v6.0 Installation Guide”. Internet Security Systems.

Internet Security Systems. (2001b). “RealSecure Product Specification”. Internet Security Systems. Retrieved from Retrived 09 December 2001, from http://documents.iss.net/literature/RealSecure/rs_ps.pdf.

Internet Security Systems. (2001c). “Getting Started Guide”. Internet Security Systems.

Internet Security Systems. (2001d). “Workgroup Manager User Guide Version 6.0”. Internet Security Systems.

Internet Security Systems. (2001e). “Network Sensor Policy Guide Version 6.0”. Internet Security Systems.

ISA-Internet Security & Acceleration Server2000 (2000). Microsoft Corporation.

ISO/IEC JTC. (1999). "Guide for the Production of PPs and STs V.0.8". ISO/IEC JTC.

Jackson, K.A. (1999). "Intrusion Detection System (IDS) Product Survey". *Los Alamos National Laboratory*.

Kendall,K. (1999). "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems". *Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degrees of Bachelor of Science in Computer Science and Engineering and Master of Engineering in Electrical Engineering and Computer Science at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY (MIT).*

Kumar,S. (1995). "Classification and Detection of Computer Intrusions". *Ph.D. Thesis, Purdue University.*

LaPadula,L.J. (1997). "Intrusion Detection for Air Force Networks". *MITRE Technical Report*.

Lippmann,R.P., Fried,D.J., Graf,I., Haines,J.W., Kendall,K., McClung,D., Weber,D., Webster,S., Wyschogrod,D., Cunningham,R., Zissman,M.A. (1998). "Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation". *DARPA Information Survivability Conference and Exposition (DISCEX*).

Lippmann,R.P., Fried,D.J., Haines,J.W., Korba,J.,.Das,K. (1999). "The 1999 DARPA Off-Line Intrusion Detection Evaluation".

Lodin,S. (1999). "Intrusion Detection Product Evaluation Criteria". *Computer Security Journal, Volume 15, Issue 2,* 1-10.

Metcalf, T.R., LaPadula,L.J. (2000). "Intrusion Detection System Requirements". *MITRE Paper*.

NASA-GB-A201. (1992). “Software Assurance Guidebook”. National Aeronaustic and Space Administration (NASA). Retrived July 24, 2001, from http://satc.gsfc.nasa.gov/assure/agb.txt.

NASA-STD-2201-93. (1992). “Software Assurance Standard”. NASA. Retrived July 24, 2001, from http://satc.gsfc.nasa.gov/assure/astd.txt.

NetSonar100 Intrusion Detection System (2001) Barbedwires Technologies

NortonAntiVirus2002. (2002). Symantec Corp.

Pointe Technology Group, Inc (2000). “Software Testing and Quality Assurance White Papers”. *Pointe Technology Group, Inc.* Retrived February 7, 2002, from http://www.itpapers.com database.

Pressman, R.S. (1994). *Software Engineering.* Europe: McGraw-Hill.

Puketza,N., Zhang,K., Chung,M., Mukherjee,B., Olsson,R.A. (1994). "A Methodology for Testing Intrusion Detection Systems". *17th National Computer Security Conference in Baltimore, MD*. 25 p.

Puketza,N., Chung,M., Mukherjee,B., Olsson,R.A. (1995). "Simulating Concurrent Intrusions for Testing IDS: Parallelizing Intrusions".

Science Applications International Corporation (SAIC). (2000). "Intrusion Detection System (IDS) System Protection Profile", "IDS Analyser Protection Profile", "IDS Sensor Protection Profile", "IDS Scanner Protection Profile". *Prepared for National*

*Security Agency (NSA) Prepared by Science Applications International Corporation (SAIC) Center for Information Security Technology, Draft.* Retrived May 2, 2001, from http://www.iatf.net/protection_profiles/intrusion.cfm.

Schepers,L. (1998). "Network Versus Host Based Intrusion Detection". *Information Security Technical Report, Volume 3, No: 4*, 32-42.

Schultz,D.J. (2000.) "A Comparison of Five Approaches to Software Development". Retrived February 8, 2002, from http://sel.gsfc.nasa.gov/website/research/tech-study/daves-white.pdf.

Sommerville,I. (1995) *Software Engineering.* Addison-Wesley.

Stoneburner,G. (2000). "COTS Security Protection Profile - Operating Systems". United States Department of Commerce, National Institute of Standards and Technology. Retrived Jan 10, 2001 from www.commoncriteria.org.

The Software Engineering Institute. (2002). "Three Tier Software Architectures". The Software Engineering Institute, Carnegie Mellon University. Retrived April 22, 2002, from http://www.sei.cmu.edu/str/descriptions/threetier.html.

Wallace, D.R., Peng, W.W., Ippolito, L.M. (1994) "Software Quality Assurance: Documentation and Reviews" U.S. DEPARTMENT OF COMMERCE Technology Administration National Institute of Standards and Technology Computer Systems Laboratory.

Webster S.E. (1998). "The Development and Analysis of Intrusion Detection Algorithms". *Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degrees of Bachelor of Science in Computer Science and Engineering and Master of Engineering in Electrical Engineering and Computer Science at the MIT*.

Williams,J.R., Schaefer,M., Landoll,D.J. (1998). "Pretty Good Assurance". *Arca Systems, Inc.* Retrived February 13, 2002 from http://www.arca.com/projects/assurance.html.

# APPENDIX A

## IDS ASSESSMENT TABLE

In this appendix an IDS assessment table is introduced. Similar tables are given in (ICSA Labs, 1999) and (Jackson, 1999), but here the difference is that this table is based on the model which is explained in Chapter three. The requirements of IDS are summarized, grouped and included according to the IDS components introduced in the IDS model (Chapter 3).

The existing components of IDS are checked in the first row of table. Since different IDSs may have these modules with different names according to the vendor or the product, the available components of IDS will be checked according to their functionality. The user of the table will fill the empty boxes by checking them if the individual property exists in the IDS.

Then in each subsection of the table, IDS modules are examined according to the requirements of modules. As those requirements are mentioned before, they won't be explained here again.

Table A.1: IDS Assessment Table.

| **IDS name:** | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Existing Modules** | | | | | | | | | | | | | | | | | | | |
| GUI | | Management Server | | Database Server | | Collector | | Analyzer | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| **GUI** | | | | | | | | | | | | | | | | | | | |
| **Deployment** | | **Windows** | | | | | | | | | | | | | | | | | |
| remote | local | IDS modules | | policies | | configuration | | detected intrusions | | | reports | | | other | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| **Management Server** | | | | | | | | | | | | | | | | | | | |
| **Response** | | **Functions (**Key: [broad, limited, none]) | | | | | | | | | | | | | | **Deployment** | | | |
| active | passive | intrusion definition | | | response | | | audit record definition | | | report generation | | | encryption options | | | remote | local | |
| | | | | | | | | | | | | | | | | | | | |
| **Database Server** | | | | | | | | | | | | | | | | | | | |
| **Databases** | | | | | | | | | | | | | | | | **Architecture** | | | |
| policy | | asset | | audit | | intrusion | | | | | | | | | | local | | distributed | |
| | | | | | | | | | | | | | | | | | | | |
| **Collector** | | | | | | | | | | | | | | | | | | | |
| **Information Source** | | | | | | | | | | | | | | | | **Architecture** | | | |
| network | O/S | application | | file system | | other | | | | | | | | | | local | | distributed | |
| | | | | | | | | | | | | | | | | | | | |
| **Analyzer** | | | | | | | | | | | | | | | | | | | |
| **Method** | | | | | | | | | | | | | | | | **Analysis** | | | |
| anomaly detection | | misuse detection | | | | other | | | | | | | | | | batch | | real-time | |
| | | | | | | | | | | | | | | | | | | | |

# APPENDIX B

## COVERAGE OF SOFTWARE ENGINEERING STANDARDS

The purpose of table B.1 is to provide the coverage of the respective three approaches to software engineering, IEEE Software Engineering (SWE) Standards, ISO/IEC 12207 Information technology-Software life cycle processes, ISO 9000 quality standards and ISO 9000-3, Guidelines for the application of ISO 9001 to the development, supply, and maintenance of software.

Table B.1: Software Engineering Standards Table (Schultz, 2000).

| **Subject Area** | **IEEE SWE Standards** | **ISO/IEC 12207** | **ISO 9001, 9000-3** |
|---|---|---|---|
| Program Management | X | X | X |
| S/W Life Cycle Processes | X | X | |
| S/W Acquisition | X | X | |
| Contract Preparation | X | X | |
| Contract Review | X | | X |
| System Engineering | X | X | |
| System Requirements | X | X | |
| Software Requirements | X | X | |
| Software Design | X | X | X |
| Implementation | X | X | X |
| System Test | X | X | X |
| Operations | X | X | |
| Maintenance | X | X | |
| Quality Management | | | X |
| Quality Assurance | X | X | X |
| Configuration Management | X | X | X |
| Reuse | X | | |
| Documentation | X | X | X |
| Measurement | X | | X |
| Training | | X | X |

# APPENDIX C

# COMMON CRITERIA SECURITY ASSURANCE REQUIREMENTS

The security assurance components drawn from CC Part 3: Security Assurance Requirements are restated for EAL2. These requirements are directly quoted from Common Criteria: Part 3 (Common Criteria Org. 1999c) with out any modification.

A set of assurance elements is provided for each assurance component. An assurance element is a security requirement, which if further divided, would not yield a meaningful evaluation result. It is the smallest security requirement recognised in the CC. Each assurance element is identified as belonging to one of the three sets of assurance elements (Common Criteria. Org. 1999c):

a) Developer action elements: the activities that shall be performed by the developer. This set of actions is further qualified by evidential material referenced in the following set of elements. Requirements for developer actions are identified by appending the letter "D" to the element number.

b) Content and presentation of evidence elements: the evidence required, what the evidence shall demonstrate, and what information the evidence shall convey. Requirements for content and presentation of evidence are identified by appending the letter "C" to the element number.

c) Evaluator action elements: the activities that shall be performed by the evaluator. This set of actions explicitly includes confirmation that the requirements prescribed in the content and presentation of evidence elements have been met. It also includes explicit actions and analysis that shall be performed in addition to that already performed by the developer. Implicit evaluator actions are also to be performed as a result of developer action elements which are not covered by content and presentation of evidence requirements. Requirements for evaluator actions are identified by appending the letter "E" to the element number.

The developer actions and content and presentation of evidence define the assurance requirements that are used to represent a developer's responsibilities in demonstrating assurance in the TOE security functions. By meeting these requirements,

the developer can increase confidence that the TOE satisfies the functional and assurance requirements of a PP or ST (Common Criteria Org. 1999c).

The evaluator actions define the evaluator's responsibilities in the two aspects of evaluation. The first aspect is validation of the PP/ST, in accordance with the classes APE and ASE in clauses 4 and 5. The second aspect is verification of the TOE's conformance with its functional and assurance requirements. By demonstrating that the PP/ST is valid and that the requirements are met by the TOE, the evaluator can provide a basis for confidence that the TOE will meet its security objectives (Common Criteria Org. 1999c).

**Assurance Configuration Management CAPabilities.2: Configuration items**

*Developer action elements:*

| | |
|---|---|
| ACM_CAP.2.1D | The developer shall provide a reference for the TOE. |
| ACM_CAP.2.2D. | The developer shall use a CM system. |
| ACM_CAP.2.3D | The developer shall provide CM documentation. |

*Content and presentation of evidence elements:*

| | |
|---|---|
| ACM_CAP.2.1C | The reference for the TOE shall be unique to each version of the TOE. |
| ACM_CAP.2.2C | The TOE shall be labeled with its reference. |
| ACM_CAP.2.3C | The CM documentation shall include a configuration list. |
| ACM_CAP.2.4C | The configuration list shall describe the configuration items that comprise the TOE. |
| ACM_CAP.2.5C | The CM documentation shall describe the method used to uniquely identify the configuration items. |
| ACM_CAP.2.6C | The CM system shall uniquely identify all configuration items. |

*Evaluator action elements:*

| | |
|---|---|
| ACM_CAP.2.1E | The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence. |

**Assurance Delivery and Operation DELivery.1: Delivery procedures**

*Developer action elements:*

ADO_DEL.1.1D The developer shall document procedures for delivery of the TOE or parts of it to the user.

ADO_DEL.1.2D The developer shall use the delivery procedures.

*Content and presentation of evidence elements:*

ADO_DEL.1.1C The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to a user's site.

*Evaluator action elements:*

ADO_DEL.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

**ADO Installation, Generation and Start-up.1: Installation, generation, and start-up procedures**

*Developer action elements:*

ADO_IGS.1.1D The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.

*Content and presentation of evidence elements:*

ADO_IGS.1.1C The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE.

*Evaluator action elements:*

ADO_IGS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS.1.2E The evaluator shall determine that the installation, generation, and start-up procedures result in a secure configuration.

**Assurance DeVelopment Functional SPecification.1: Informal functional specification**

*Developer action elements:*

ADV_FSP.1.1D The developer shall provide a functional specification.

*Content and presentation of evidence elements:*

ADV_FSP.1.1C The functional specification shall describe the TSF and its external interfaces using an informal style.

ADV_FSP.1.2C The functional specification shall be internally consistent.

ADV_FSP.1.3C The functional specification shall describe the purpose and method of use of all external TSF interfaces, providing details of effects, exceptions and error messages, as appropriate.

ADV_FSP.1.4C The functional specification shall completely represent the TSF.

*Evaluator action elements:*

ADV_FSP.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.1.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the TOE security functional requirements.

**ADV High-Level Design.1: Descriptive high-level design**

*Developer action elements:*

ADV_HLD.1.1D The developer shall provide the high-level design of the TSF.

*Content and presentation of evidence elements:*

ADV_HLD.1.1C The presentation of the high-level design shall be informal.

ADV_HLD.1.2C The high-level design shall be internally consistent.

ADV_HLD.1.3C The high-level design shall describe the structure of the TSF in terms of subsystems.

ADV_HLD.1.4C The high-level design shall describe the security functionality provided by each subsystem of the TSF.

ADV_HLD.1.5C The high-level design shall identify any underlying hardware, firmware, and/or software required by the TSF with a presentation of the functions provided by the supporting protection mechanisms implemented in that hardware, firmware, or software.

ADV_HLD.1.6C The high-level design shall identify all interfaces to the subsystems of the TSF.

ADV_HLD.1.7C The high-level design shall identify which of the interfaces to the subsystems of the TSF are externally visible.

*Evaluator action elements:*

ADV_HLD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD.1.2E The evaluator shall determine that the high-level design is an accurate and complete instantiation of the TOE security functional requirements.

**ADV Representation of CoRrespondence.1: Informal correspondence demonstration**

*Developer action elements:*

ADV_RCR.1.1D The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.

*Content and presentation of evidence elements:*

ADV_RCR.1.1C For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation.

*Evaluator action elements:*

ADV_RCR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

**Assurance Guidance Documents ADMinistrator guidance.1: Administrator guidance**

*Developer action elements:*

AGD_ADM.1.1D The developer shall provide administrator guidance addressed to system administrative personnel.

*Content and presentation of evidence elements:*

AGD_ADM.1.1C The administrator guidance shall describe the administrative functions and interfaces available to the administrator of the TOE.

AGD_ADM.1.2C The administrator guidance shall describe how to administer the TOE in a secure manner.

AGD_ADM.1.3C The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.

AGD_ADM.1.4C The administrator guidance shall describe all assumptions regarding user behavior that are relevant to secure operation of the TOE.

AGD_ADM.1.5C The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate.

AGD_ADM.1.6C The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_ADM.1.7C The administrator guidance shall be consistent with all other documentation supplied for evaluation.

AGD_ADM.1.8C The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator.

*Evaluator action elements:*

AGD_ADM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

**AGD UseR guidance.1: User guidance**

*Developer action elements:*

AGD_USR.1.1D The developer shall provide user guidance.

*Content and presentation of evidence elements:*

AGD_USR.1.1C The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE.

AGD_USR.1.2C The user guidance shall describe the use of user-accessible security functions provided by the TOE.

AGD_USR.1.3C The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

AGD_USR.1.4C The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behavior found in the statement of TOE security environment.

AGD_USR.1.5C The user guidance shall be consistent with all other documentation supplied for evaluation.

AGD_USR.1.6C The user guidance shall describe all security requirements for the IT environment that are relevant to the user.

*Evaluator action elements:*

AGD_USR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

**Assurance TEsts COVerage.1: Evidence of coverage**

*Developer action elements:*

ATE_COV.1.1D The developer shall provide evidence of the test coverage.

*Content and presentation of evidence elements:*

ATE_COV.1.1C The evidence of the test coverage shall show the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification.

*Evaluator action elements:*

ATE_COV.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

**ATE FUNctional tests.1: Functional testing**

*Developer action elements:*

ATE_FUN.1.1D The developer shall test the TSF and document the results.

ATE_FUN.1.2D The developer shall provide test documentation.

*Content and presentation of evidence elements:*

ATE_FUN.1.1C The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results.

ATE_FUN.1.2C The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.

ATE_FUN.1.3C The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests.

ATE_FUN.1.4C The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE_FUN.1.5C The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified.

*Evaluator action elements:*

ATE_FUN.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

**ATE INDependent testing.2: Independent testing – sample**

*Developer action elements:*

ATE_IND.2.1D The developer shall provide the TOE for testing.

*Content and presentation of evidence elements:*

ATE_IND.2.1C The TOE shall be suitable for testing.

ATE_IND.2.2C The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

*Evaluator action elements:*

ATE_IND.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.2.2E The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified.

ATE_IND.2.3E The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

**Assurance Vulnerability Assessment Strength Of TOE security Function.1: Strength of TOE security function evaluation**

*Developer action elements:*

AVA_SOF.1.1D The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.

*Content and presentation of evidence elements:*

AVA_SOF.1.1C For each mechanism with a strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level defined in the PP/ST.

AVA_SOF.1.2C For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of

function metric defined in the PP/ST.

*Evaluator action elements:*

AVA_SOF.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_SOF.1.2E The evaluator shall confirm that the strength claims are correct.

**AVA VuLnerability Analysis.1: Developer vulnerability analysis**

*Developer action elements:*

AVA_VLA.1.1D The developer shall perform and document an analysis of the TOE deliverables searching for obvious ways in which a user can violate the TSP.

AVA_VLA.1.2D The developer shall document the disposition of obvious vulnerabilities.

*Content and presentation of evidence elements:*

AVA_VLA.1.1C The documentation shall show, for all identified vulnerabilities, including those identified in Appendix A of ALFPP v1.d.1, that the vulnerability cannot be exploited in the intended environment for the TOE.

*Evaluator action elements:*

AVA_VLA.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.1.2E The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure obvious vulnerabilities have been addressed.

# APPENDIX D

# PROPOSED COMMON CRITERIA INTRUSION DETECTION CLASS

## Class FID: Intrusion Detection

Intrusion detection class involves collecting, recognising, and analysing information related to IT system activities (i.e. activities controlled by the TSP). The collected data from IT system sources can be examined to determine the past, present and future intrusions occurance.

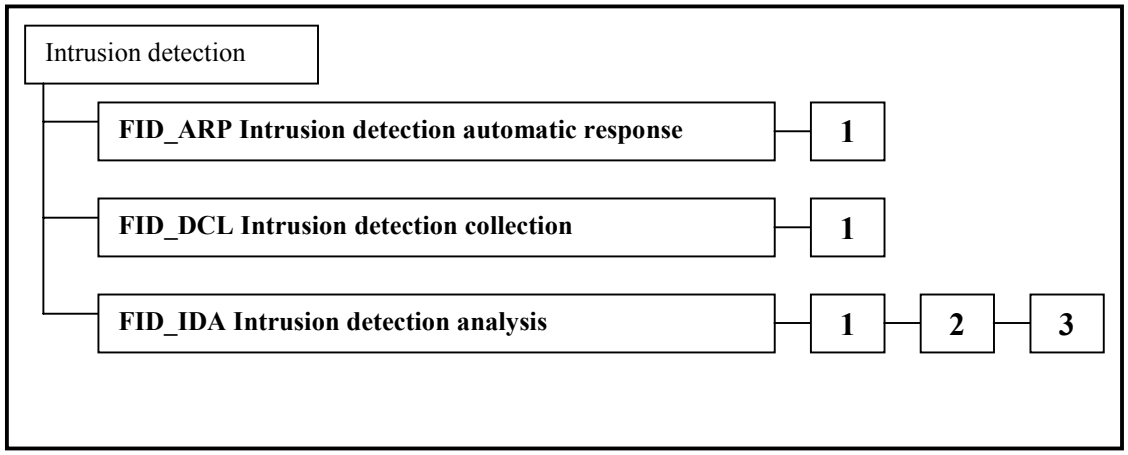


Figure D.1: Intrusion detection class decomposition.

### Intrusion detection automatic response (FID_ARP)

*Family behaviour:*

This family defines the response to be taken in case of detected events indicative of an intrusion.

Component levelling:

| | |
|---|---|
| **FID_ARP Intrusion detection automatic response** | **1** |

At FID_ARP.1 Security alarms, the TSF shall take actions in case an intrusion is detected.

Management: FID_ARP.1

The following actions could be considered for the management functions in FMT:

a) the management (addition, removal, or modification) of actions.

Audit: FID_ARP.1

The following actions should be auditable if FID_GEN Security audit data generation is included in the PP/ST:

a) Minimal: Actions taken due to imminent intrusions.

**FID_ARP.1 Security alarms**

Hierarchical to: No other components.

FID_ARP.1.1 The TSF shall take [assignment: *list of the least disruptive actions*] upon detection of an intrusion.

Dependencies: FID_IDA.1 Intrusion analysis

**Intrusion detection collection (FID_DCL)**

Family behaviour:

This family defines requirements for collecting the occurrence of security relevant events and/or data that take place under TSF control and IT system. This family identifies the level of collecting, enumerates the types of events that shall be collectable by the TSF.

Component levelling:

| | |
|---|---|
| **FID_DCL Intrusion detection collection** | **1** |

FID_DCL.1 Data collection defines the level of collectable events, and specifies the list of data that shall be collected.

Management: FID_IDC.1

The following actions could be considered for the management functions in FMT:

a) maintenance (deletion, modification, addition) of the collectable events.

Audit: FID_IDC.1

The following actions should be auditable if FID_GEN Security audit data generation is included in the PP/ST:

a) Minimal: Enabling and disabling of any of the collection mechanisms.

**FID_DCL.1 Data Collection**

Hierarchical to: No other components.

FID_DCL.1.1 The TSF shall be able to collect [assignment: *other specifically defined possible actions such as store in the database and/or send for the analysis*] [assignment*: specially defined IT system event and/or data such as network packet data*] from the IT system resources.

Dependencies: No dependencies

**Intrusion detection analysis (FID_IDA)**

*Family behaviour:*

This family defines requirements for automated means that analyse system activity on collected system data looking for possible or real intrusions. This analysis may work in support of intrusion detection, or automatic response to an intrusion.

The actions to be taken based on the detection can be specified using the FID_ARP family as desired.

Component levelling:


In FID_IDA.1 Potential violation analysis, basic threshold detection on the basis of a rule set is required.

In FID_IDA.2 Simple attack heuristics, the TSF shall be able to detect the occurrence of signature events that represent a significant intrusion. This search for signature events occurs in real-time.

In FID_IDA.3 Complex attack heuristics, the TSF shall be able to represent and detect multi-step intrusion scenarios. The TSF is able to compare system events (possibly performed by multiple individuals) against event sequences known to represent entire intrusion scenarios. The TSF shall be able to indicate when a signature event or event sequence is found that indicates an intrusion.

Management: FID_IDA.1

The following actions could be considered for the management functions in FMT:

a) maintenance of the rules by (adding, modifying, deletion) of rules from the set of rules.

Management: FID_IDA.2

The following actions could be considered for the management functions in FMT:

a) maintenance (deletion, modification, addition) of the subset of system events.

Management: FID_IDA.3

The following actions could be considered for the management functions in FMT:

a) maintenance (deletion, modification, addition) of the subset of system events;

b) maintenance (deletion, modification, addition) of the set of sequence of system events.

Audit: FID_IDA.1, FID_IDA.2, FID_IDA.3.

The following actions should be auditable if FID_GEN Security audit data generation is included in the PP/ST:

a) Minimal: Enabling and disabling of any of the analysis mechanisms;

b) Automated responses performed by the tool.

c) Other intrusion related information belonging to detected intrusion such as intrusion type and priority.

**FID_IDA.1 Intrusion detection analysis**

Hierarchical to: No other components.

FID_IDA.1.1 The TSF shall be able to apply a set of rules in monitoring the collected event data and based upon these rules indicate an intrusion.

FID_IDA.1.2 The TSF shall enforce the following rules for monitoring collected data:

a) Accumulation or combination of [assignment: *subset of defined collectable event data*] known to indicate an intrusion;

b) [assignment: *any other rules*].

Dependencies: FID_COLLECTION.1 Data collection

**FID_IDA.2 Simple attack heuristics**

Hierarchical to: FID_IDA.1

FID_IDA.2.1 The TSF shall be able to maintain an internal representation of the following signature events [assignment: *a subset of system events*] that may indicate an intrusion.

FID_IDA.2.2 The TSF shall be able to compare the signature events against the collected system data discernible from an examination of [assignment: *the information to be used to determine collected system data*].

FID_IDA.2.3 The TSF shall be able to indicate an imminent intrusion when a system event is found to match a signature event that indicates an intrusion.

Dependencies: No dependencies

**FID_IDA.3 Complex attack heuristics**

Hierarchical to: FID_IDA.2

FID_IDA.3.1 The TSF shall be able to maintain an internal representation of the following event sequences of known intrusion scenarios [assignment: *list of sequences of system events whose occurrence are representative of known penetration scenarios*] and the following signature events [assignment: *a subset of system events*] that may indicate an intrusion.

FID_IDA.3.2 The TSF shall be able to compare the signature events and event sequences against the collected system data discernible from an examination of [assignment: *the information to be used to determine collected system data*].

FID_IDA.3.3 The TSF shall be able to indicate an imminent intrusion when system activity is found to match a signature event or event sequence that indicates an intrusion.

Dependencies: No dependencies