Securing Network Connected Applications with Proposed Security Models

Mustafa Tahir
Dimitrios Konstantaras

Report 08035
ISSN 1650-2647
ISRN VXU/MSI/DA/E/--08035/--SE
Mustafa Tahir  Dimitrios Konstantaras

Securing Network Connected Applications with Proposed Security Models

Bachelor’s Thesis

Computer Science

2008
Abstract

In today’s society, serious organizations need protection against both internal and external attacks. There are many different technologies available that organizations can incorporate into their organization in order to enhance security for their networking applications. Unfortunately, security is way to often considered as an afterthought and therefore implemented as an external part of the applications. This is usually performed by introducing general security models and technologies.

However, an already developed, well structured and considered security approach – with proper implementation of security services and mechanisms – different security models can be used to apply security within the security perimeter of an organization. It can range from built into the application to the edge of a private network, e.g. an appliance. No matter the choice, the involved people must possess security expertise to deploy the proposed security models in this paper, that have the soul purpose to secure applications.

By using the Recommendation X.800 as a comparison framework, the proposed models will be analyzed in detail and evaluated of how they provide the security services concerned in X.800. By reasoning about what security services that ought to be implemented in order to prevent or detect diverse security attacks, the organization needs to carry out a security plan and have a common understanding of the defined security policies.

An interesting finding during our work was that, using a methodology that leads to low KLOC-values results in high security, though low KLOC-values and high security go hand-in-hand.

Key-words: Security service, security mechanism, security protocol, security policy, X.800 standard, attacks, application proxy, firewall, Virtual Private Network (VPN), plugin, filter, Team Software Process (TSP), IPsec, Common Criteria (CC), DMZ.
# Table of Contents

1 **Introduction**  
1.1 Problem Definition  
1.2 Motivation  
1.3 Method  
1.4 Restrictions  

2 **Theory**  
2.1 Common Criteria  
2.2 X.800, the OSI Security Architecture  
  2.2.1 OSI Reference Model  
  2.2.2 Security Attacks  
  2.2.3 Security Services  
  2.2.4 Security Mechanisms  
2.3 A Model For Network Security  
2.4 Firewalls  
  2.4.1 Application-proxy  
2.5 Virtual Private Networks (VPN)  
  2.5.1 IP Security (IPsec)  
  2.5.2 Site-to-site VPN  
  2.5.3 Remote-Access  
  2.5.4 Demilitarized Zone (DMZ)  

3 **Proposed Security Models**  
3.1 VPN Design  
  3.1.1 Proposed Topology  
3.2 Application-Proxy Design  
3.3 Filter  
3.4 Plugin Design  
  3.4.1 Example  
3.5 Built-In To The Software  

4 **Analysis**  
4.1 Virtual Private Networks  
4.2 Application-Proxy  
4.3 Filter  
4.4 Plugin  
4.5 Built-In To The Software  

5 **Discussion**  
5.1 Conclusions  
5.2 Future Work and Improvements  

**References**  

**Appendices**  

A **Security Analysis**  
  A.1 Risk Analysis  
  A.2 Security Policy  
  A.3 Security Plan  

B **RADIUS**  

C **Pervasive Security Mechanisms**  

D **Provided Security Services**
## List of Figures

<table>
<thead>
<tr>
<th></th>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>OSI Reference model</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Model for Network Security</td>
<td>9</td>
</tr>
<tr>
<td>2.3</td>
<td>Transport mode</td>
<td>11</td>
</tr>
<tr>
<td>2.4</td>
<td>Tunnel mode</td>
<td>12</td>
</tr>
<tr>
<td>3.1</td>
<td>VPN Design</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>Application-Proxy Design</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>Filter</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>Plugin design</td>
<td>20</td>
</tr>
</tbody>
</table>
1 INTRODUCTION

Network security is a critical issue and a challenge to professional system developers in means of protection against diverse attacks aimed at any resource that is of interest to the attacker. Over the years, with the increasing number of computer systems connected to the global Internet, even the average users must be aware of the external threats and therefore also take some kind of precautions in order to protect themselves from these threats. However, from a business’ or an organization’s point of view, security assessments are of a greater importance and must be acknowledged and valued in order to form the security policies that are associated with an organization or a business.

This paper deals specifically with network security and barely with data security. It is therefore important to emphasize that security is often considered as an afterthought when providing security for an application. This is generally performed by implementing security as an external part of the application, by securing the information (network data) transmitted by the applications, with general security models and technologies. This is the common approach used today to secure applications.

A less common approach is to incorporate security as a software process in the development life-cycle of the applications. This means that security is implemented (built-in) in to the software. Not only does this approach include data security, but also network security; the latter, generally through implementing different security services.

Throughout this paper, we will provide the reader with different available security models, with their soul purposes to secure applications.

1.1 PROBLEM DEFINITION

Many of the software systems that we know of today have some kind of security already embedded in the application, where the software engineers spend more time on making the system secure. With an already developed, well structured and considered security approach to secure networking applications – the question that arises is where the security should be applied within the security perimeter? And based on where the security is applied, in which way does it impact on the application? Furthermore, how do all the involved people get affected by different security deployments? Consequently, do different security deployments result in that the developers spend less time on implementing the security in the application?

By presenting our security models – to attempt to secure an application in a networking system – we are able to emphasize different security issues that ought to be considered in advance by all the involved people. We are therefore assuming that applications that need to be secured do not preserve any existing security. Hence, with the proposed security models, we will at least try to assure security by means of X.800.

We define our problem domain as:

1. In which ways can you secure an application?

2. How can you apply a security component, a component consisting of software, hardware or a combination of both, that provide some kind of security features to applications that do not preserve any or only some features?

3. Finally, what distinguishes the various security models in terms of securing applications?

1.2 MOTIVATION

The motivation with this thesis is to describe the general security features that must be implemented in order to make an application secure. However, the main motivation has derived from the considerations of software systems that do not have any security features implemented. Therefore, our purpose is to evaluate how to develop and construct a security component (a software component) that can easily be applied to an already existing system, thus the programmers can spend more time on the functionality and hand over the responsibility of security issues to the component.

Further, this thesis is also written to provide different security models, in means of securing networking applications. These models are proposed to serve as an implementation and deployment guidance for anyone who wishes to introduce security into their applications.

1.3 METHOD

To understand the security aspects of each model, a thorough description is given as a theory basis on security services and mechanisms that are clearly defined in X.800 standard. The theory will eventually,
throughout the thesis, be used when analyzing the models in more detail and serve as a comparison framework when evaluating if the models provide the adequate security services concerned in X.800 standard when securing networking applications.

The models will also be evaluated with respect to the security requirements expressed by Common Criteria, that is, if the provided model can achieve an adequate level of security.

1.4 Restrictions

This thesis is intended for people that are familiar with the general concepts of network security and networking. We only consider the concepts of network security for the proposed security models. Where security in means of hardware (e.g. network equipment) is only mentioned briefly. Therefore, description of many terms and concepts are left out, purposely, since these are assumed to be known by the reader.
2 Theory

This chapter will serve as a basis and an overview to the security issues and aspects that will be addressed throughout the thesis. We will introduce you to terms and concepts about network security to facilitate the comprehension and the elaboration of this paper’s contents.

2.1 Common Criteria

Common Criteria (CC) is an international standard aimed towards computer security. The CC is used to evaluate different security products and systems and can also serve as guidance when developing secure products [2]. The level of security for different technology products can be assessed with CC, though the security requirements are thoroughly expressed and the criteria of the product in question are clearly defined by CC. The evaluation process is taken out by focusing on a part of a product or a system and this part is called Target Of Evaluation (TOE). This includes, security threats, requirements, summarized specification of security functions and the assurance measure (e.g. if the TOE meets the claimed security). The assurance measures are requirements the system or the product has to meet and these are specified in levels called Evaluation Assurance Levels (EAL0-7). EAL0 has no assurance at all and in contrast, EAL7 has the highest assurance level. These are then the inputs to the Security Target (ST) that the evaluators can use as a rationale when evaluating the product or the system. One or more reports about the evaluation results are documented to confirm if the ST is satisfied with respect to TOE. Revision may be required of the TOE if any vulnerabilities arise, thus re-evaluation may be inevitable [3].

Generally, the rationale of CC copes with computer system users or product users that state their security requirements – vendors can, with regard to the security attributes, implement or make assumptions that their products or systems satisfy the specified security requirement. Further tests, in special testing laboratories, are then taken out in order to evaluate if their implementation or their claims actually fulfill the requirements. These security requirements are defined in a Protection Plan (PP), that is usually created and defined by potential consumers or developers of a product, which the vendor aims to comply with these requirements when they are creating their products or systems [2] [3]. It is worth mention, that the products are evaluated against the PP, therefore the same requirements as in the PP ought to be stated in the documented reports produced as the output of a ST. In other words, the PP and ST serves as a basis for the TOE.

When a system is successfully tested and evaluated with respect to CC – the system or the product is provided with a certificate. This gives it trustworthiness in a more standardized manner. However, even though a certificate is provided showing that the system or the product fulfills the CC’s requirements, there is no detailed description or guidance on how to achieve these requirements [4]. Therefore, the recommendation X.800 plays an important role of guidance in a more technical and systematic description of security development.

2.2 X.800, The OSI Security Architecture

With the various security policies defined and established through the elaborative analysis described in Appendix A, through carefully assessments of the appropriate level of security needed by an organization. The requirements to achieve that security must be defined in a systematic way by the manager that is responsible for security. Also the approaches to satisfy those requirements must be characterized. This systematic approach is however not an easy task for the manager to define in a world of interconnected heterogeneous computer systems [5].

Recommendation X.800, Security Architecture for Open Systems Interconnect (OSI), provides the security managers with such systematic approach. With this approach as a foundation (with guidelines and constraints), the organization can establish its own framework of coordinated security architecture development within the context of OSI to provide security. In other words, the general architectural elements that is defined in X.800 covers the necessary security aspects and in spite of its abstract description, it provides an overall description of any system’s structure of how to fulfill that system’s security requirements. It will therefore still be useful to achieve the objectives defined in the security policies of the organization [5][6].

The usefulness of the recommendation X.800, when applying the security-related architectural elements within the organization’s framework, derives from its establishment as an international standard. This standard has been used as a primary architectural model by many communication and computer vendors in their development process of security features for their products and services. With the usage of this abstract unified model, they have thereby overcome the compounded problems of heterogeneity in the scope of secure communication. Architectural elements are basically security services/mechanisms and other properties defined in X.800[5].
The systematic approach defined in the security architecture for OSI, describes security services and security mechanisms to ensure an adequate security. These services and mechanisms are placed in a relation to the appropriate layer within the OSI Reference Model. To satisfy the security requirements defined by the organization, usually, the basic services and mechanisms provided by the OSI security architecture are not enough. Consequently, in practice, a combination with additional non-OSI services and mechanism must be considered. Due to the fact that, within the Reference Model, each layer provides a well defined interface to a layer above and respectively below. Thus the implementation of any non-OSI services/mechanisms, that is needed within the organization’s framework, can be made fairly easy and flexible and still be acceptable by the OSI standard. The OSI Reference Model will we briefly described, in section 2.2.1.

To summarize, the OSI security architecture is the underlying technical description of the objective and the policy. Furthermore, the services correspond to the implementation of the policies and to achieve the required level of security, the mechanisms are incorporated into an appropriate layer within the Reference Model to employ those services. Thus, with the security mechanisms, which implement the security services, an organization can build up the adequately needed security [7][6][5][8].

2.2.1 OSI Reference Model

The OSI reference model describes each of the seven layers that are used between computers which are connected to networks. When two computers communicate through some application, the communication is passed down the layers, through a network and up through the layers on the receiving side. Each layer can communicate with the adjacent layer, for example the transport layer can communicate with the session layer and the network layer, but not directly with the application layer (see figure 2.1).

The layers can be grouped in to two categories, namely Application and Data Transport. This categorization is made because the three highest layers mainly involve application issues and are implemented in software, whereas the four lower ones generally handle data transport issues and are implemented in hardware as well as in software. A layer provides some functionality in addition to the services provided by a lower layer, and the layer above can use those services [9]. The different layers are the following:

**APPLICATION LAYER** - applications that use the network (more or less every application nowadays).

**PRESENTATION LAYER** - encodes data in a system-independent format (encryption, compression or conversion of data).

**SESSION LAYER** - handles the establishment and termination of connections between hosts.

**TRANSPORT LAYER** - handles packet loss and retransmissions to ensure reliable communication (TCP).

**NETWORK LAYER** - navigation through interconnected networks (through routers, switches, etc).
**DATA LINK LAYER** - transfers data with the use of the underlying layer and ensures that the right packets arrive at the correct destination.

**PHYSICAL LAYER** - implementation of different network technologies (LAN, WAN, cables, modems, topologies) and how data is sent over these.

The OSI reference model is not widely used and is more considered as a guidance and as an architectural model, when developing networked technologies [10, 7, 11].

### 2.2.2 Security Attacks

Before we discuss the security services and mechanism, one must perceive the concept of security attacks. These attacks can be classified into two kinds as they are in [8] namely: passive attacks and active attacks. By having this classification in mind, any kind of attack, that may threaten the organization’s security, could be prohibited with various security services. To detect, prevent or recover from an attack is solved by employing one or more security mechanisms.

To clarify the meaning of the term attack, a commonly used definition is stated in [12], which is not the same as the term threat in Definition A.1. However, this is a common misconception and we will therefore give the proper definition of an attack:

**Definition 2.1 (Attack).** "An assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system” [12].

#### Passive Attacks

Briefly, a passive attack is simply a gathering of information from the system, thus the system resources are not compromised. Still, we must emphasize the prevention of these types of attacks, though we do not want sensitive or confidential information to end up in wrong hands (e.g. an opponent who may have intentions to do harm with the obtained information). However, the obtained information is not altered whatsoever by the opponent – detecting a passive attack can be rather difficult for that matter – yet still, we can complicate the usefulness of the information simply by applying some kind of encryption mechanism on the data before transmission.

These passive attacks are generally two types, first; the **release of a message contents**, which is the reading of confidential message contents by an opponent; the second type is **traffic analysis**, which is when an opponent is observing and intercepting the pattern of transmitted information. Even though the traffic information is encrypted, an opponent can still analyze the encrypted messages and extract useful information from the patterns. Generally, the more messages an opponent have observed, the more information can be deduced from the traffic [8].

#### Active Attacks

In contrast to passive attacks, active attacks involve some kind of modification or falsification of the traffic. There are four different kinds of active attacks to take into consideration according to [8], namely: masquerade, replay, modification of messages and denial-of-service.

First, a **masquerade** is when an intruder impersonates the one end of the communication, thus might gain unauthorized access to some privileged information or resource. Second, a **replay** attack is when an opponent passively gathers data and at a later time retransmits this data and repeat some originally authorized action for his/her benefit. Third type of active attack, **modification of messages** is self explanatory – parts of the messages are modified so that the opponent can be granted unauthorized access to privileged user data – or change the packet headers in order to redirect the messages to a desired location. Forth and last type of attack is **denial-of-service**, which is when an opponent prevents the usage of a particular service provided by a server. A more severe form of denial-of-service attack is when the entire network is overloaded with messages to either decrease performance or collapse the network.

### 2.2.3 Security Services

This section describes the categorized security services that we briefly mentioned above without including any of the non-OSI services and mechanisms. All these services is however not mandatory to achieve the acceptable security, but it is still important for security manager to know that they are provided by the OSI Reference Model. Within this framework of well defined and structured services, the services are
categorized in five different categories with fourteen specific services. With appropriate combinations of these, the organization can agree on a primary architectural security design that is defined in their policy. The general security services are categorized and described here below as they are in [6] and [8].

**AUTHENTICATION** This service provides an assurance of an authentic communication. That is, if the arrived message is from the claimed source, then the recipient is assured that the communicating source entity is authentic. This also pertains to the receiver – the sender is assured that the recipient is the one that is part of the asserted authentic communicating entity, usually by authenticate itself by the use of a password.

In X.800, two more specific authentication services are defined:

- **Peer entity authentication** – Provides a confirmation that the identity of a peer entity is the valid associated entity at the initial connection, or during data transfer phase of a particular connection. This service provides an assurance that any of the associated entities are not masquerading their identity or attempting an unauthorized replay of a previous connection. Thus, the associated entity (e.g. the recipient of the sent data) is the one we intended, since this entity has to authenticate its identity.
- **Data origin authentication** – This service enables that the sender of a certain message is identified as the claimed source entity. However, this property authenticates the source entity but the certain message can still be modified or duplicated while in transit.

**ACCESS CONTROL** This involves the prevention or limitation of unauthorized access and use of resources. The rules of this service states that a resource can only be accessed by authorized user and what these users are authorized to do. This is implemented by letting first every user identify itself to get authenticated and thereby the correct privileges can be assigned to the user.

**DATA CONFIDENTIALITY** This service protects any unauthorized disclosure of the transmitted data probably through a passive attack. Furthermore, the confidentiality of the data can be categorized into four different groups where each group can be applied onto a specific layer of the OSI Reference Model. The categories are:

- **Connection Confidentiality** – Involves protection of the data on a connection.
- **Connectionless Confidentiality** – This protects the user data in each and every sent packet.
- **Selective-Field Confidentiality** – Only specific fields in the user data on a connection or in each packet.
- **Traffic-Flow Confidentiality** – This category is concerned with the protection of the traffic to prevent an opponent to perform the passive attack of traffic analysis. The prevention involves that the opponent cannot deduce information as: source, destination, frequency, length of the traffic during an ongoing communication.

**DATA INTEGRITY** This service aims to detect any of the four specific active attacks described in section 2.2.2. That is, if the transmitted data has been modified from source to destination, this service only reports that at deliberate attack has occurred and the integrity of the data may have been violated by an unauthorized entity. If this is the case – then this violation is reported to another part of the software or to an authorized user (e.g human), whom will take actions in order to recover from this violation.

- **Communication Integrity with Recovery** – This service attempts to recover from, and detect any active attack. Integrity is provided for all the messages (user data) sent and received on a stream.
- **Communication Integrity without Recovery** – As the name of this specific service implies; that is, exactly same features as the service above but there are no attempts to recover from an active attack.
• **Selective-Field Connection Integrity** – Provides integrity for specific, selected field(s) within the user data of a data block during transmission over a connection. Where we only check these specific, selected field(s), whether they have been affected by an active attack or not.

• **Connectionless Integrity** – This specific service provides for the integrity of a single data block, sent and received and will only detect if the data has been modified. Additionally, a replay detection can also be provided if required.

• **Selective-Field Connectionless Integrity** – Similarly, as for the connection-oriented specific service in means of selected fields, this service determines whether they have been modified or not and the differences are that integrity is provided only for a single data block.

**NON-REPUTATION** This service aims to provide absolute assurance that none of the communication entities (e.g. sender and receiver) will falsely deny the delivery or/and the reception of a message. To enable this assurance the non-repudiation service must take the one or/and both forms:

- **Non-repudiation with proof of origin** – When a message is sent, the receiver of the message can prove that the message actually was sent by the claimed sender. This specific service will also protect any attempts of the sender to falsely repudiate the sending of a message.

- **Non-repudiation with proof of destination** – Similarly, when a message is received, the sender of the message can be assured that the claimed destination actually received the message and any attempts to repudiate the reception of a message by the receiver, will be prevented by this specific service.

**AVAILABILITY** In [6] and [8] the notion of “availability” is not specified as one of the specific security services. Instead it is defined as a property that is associated with the specific services mentioned above. We will therefore use the notion “property” and not “service” in means of ensuring the system’s availability. Generally, whenever an authorized request by a user has been made to access and use the system or a system resource, this property will assure that these are available. Furthermore, this availability property tackles the possible problems when a denial-of-service attack is performed.

### 2.2.4 Security Mechanisms

To keep a system secure, different mechanisms must be implemented. These mechanisms can be categorized in the following categories depending on the security aspects that are to be addressed. These mechanisms describe security weaknesses and by overcoming these, a satisfying level of security can be achieved.

Table C.1 in Appendix C shows the **Pervasive Security Mechanisms**. These mechanisms do not particularly implement any of the security services mentioned above nor can they be placed in any protocol layer in the OSI Reference Model. Whereas, the table 2.1 shows the **Specific Security Mechanisms** which can be implemented in the different protocol layers in order to provide the security services described previously in section 2.2.3. These mechanisms are described in [6] and [8] as following:

### 2.3 A Model For Network Security

To grasp the concepts and terms about network security that will be used throughout this thesis, we will depict the same general model that is proposed by William Stallings in [8]. This model captures all the essential entities that are included in a rationale of network security. Let’s give a brief description about this model which is depicted in Figure 2.2.

First of all, a message is transmitted from the source (Sender) to the destination (Recipient) across some communication entity (e.g. Internet). But for this message exchange to take place, the two participants must agree on the use of some kind of communication protocols. This is where the TCP/IP protocol comes in hand. However, TCP/IP will not be discussed in this thesis, it is assumed that the reader is already familiar with this concept. The information channel is the route defined to transmit the message across the internet, whereby this coordinated communication establishment is provided by the TCP/IP protocol.

In order to secure this communication, that is, to protect the messages during transmission so an opponent would not evade the security services defined in section 2.2.3, two components must be included. These two components are defined as; **Security-related transformation** and **Secret information** in the model. The former deals with the messages to be transmitted – where some kind of encryption mechanism is used – either to make the message unreadable to the opponent or/and verify that the message was actually sent by the Sender. Whereas the latter component makes it possible to apply some kind of encryption mechanism
TABLE 2.1: Specific Security Mechanisms

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encipherment</td>
<td>Changing the data into an unreadable form with the use of mathematical algorithms. This makes the data useless to someone that has acquired the data if the decryption key is unknown.</td>
</tr>
<tr>
<td>Digital Signature</td>
<td>Adding additional data to some data in order to prove to the recipient that the sender is the one he/she claims to be.</td>
</tr>
<tr>
<td>Access Control</td>
<td>Limiting the access to resources only to authorized users and thus preventing others from accessing the resources.</td>
</tr>
<tr>
<td>Data Integrity</td>
<td>Protects data units from being modified during transmission in a connection.</td>
</tr>
<tr>
<td>Authentication Exchange</td>
<td>Used when entities want to communicate and a password is used for the identification between them. This mechanism can be implemented slightly different depending on the level of trustfulness between the entities.</td>
</tr>
<tr>
<td>Traffic Padding</td>
<td>Used to create fake traffic to confuse network traffic analysts. Encrypting the fake data units, further confuses traffic analysts.</td>
</tr>
<tr>
<td>Routing Control</td>
<td>Used whenever sensitive data must be transferred over a secure path. Then the path offered is ensured to have a higher level of security.</td>
</tr>
<tr>
<td>Notarization</td>
<td>A trusted third party is used to confirm the exchanged information between two communicating entities.</td>
</tr>
</tbody>
</table>

onto the message along with a secret key for encryption and decryption of the message. This secret key corresponds to the "secret information" displayed in Figure 2.2. Furthermore, this key is assumed to be known only by the two participants according to the model.

The trusted third party in the figure plays its part as a distributor of the secret key mentioned above. Both participants establish a secure communication with this trusted third party in order to keep an opponent from accessing the secret information. This trusted third party might be useful if the two participants are located geographically on distant places. Else, if the participants were located close to each other and were able to meet in person, the exchange of the secret key would be easy and probably the most secure way.

With what we describe so far and with the help of the figure 2.2, four different tasks must be carried out whenever a security service is to be designed.

1. The secure transformation must be performed by an algorithm which is carefully designed in order to make it difficult for an opponent to breach.
2. The algorithm must be supplied with secret information which should be able to be generated.
3. The secret information must be able to be distributed, in a secure manner, to the participants by some method.
4. A specific protocol needs to be agreed upon and adopted by the participants to make proper use of the algorithm in (1) and the secret information in (2).

2.4 FIREWALLS

A firewall is a technology implemented in hardware and software and is usually placed between the internal network of an organization and the Internet. A firewall’s basic task is to filter out unwanted and most often incoming traffic, whereas outgoing traffic is usually allowed regardless of the kind of traffic it is. Thus, the firewall provides security to a certain level. This technique is called packet filtering. Each packet’s header fields is examined, and based on the contents, the packet filter determines which packets can pass on to the next network and which cannot [13][14].

Firewalls generally control access and enforce security policies by using four techniques, namely; Service Control, Direction Control, User Control and Behavior Control. Service Control is used to decide which services, both incoming and outgoing, that the hosts can access. Direction Control, as the name implies, is used to decide the direction in which services can be requested and the traffic can flow. User
Control is used to determine if a user has privileges for a particular service in order to gain access to the service. Behavior Control is used to control the way a host use a certain service [8].

2.4.1 Application-proxy

While a general firewall examines incoming and outgoing packets by inspecting the headers of each packet – an application-proxy takes it one step further – by examining the actual contents in the payload (data part) of each packet, which makes it difficult for malicious code to pass through and exploit the weaknesses of the target application.

Another important aspect of a proxy is that there is no open connection during the processing of the packets. More precise, when a packet arrives, it is copied and the connection with the source is terminated. The packet’s contents is examined against the stated criteria, and if the packet passes the control then the proxy creates a new packet, establishes a new connection with the destination host, and sends the packet. Thus, neither the source nor the destination application is aware of the proxy’s existence on the network – making the proxy transparent to the application – by acting as the destination host. Since there is no direct connection between the source and the destination, the application proxy provides high security to server applications [15]. This high security is also a result of breaking down the packets to their building blocks and create new ones, assuming that they fulfill the stated criteria. If the application proxy firewall detects any malicious code the packet is discarded and the connection is terminated.

Furthermore, these type of firewalls requires that a proxy must be implemented for each specific application, and the application must be configured with the address of the proxy. If the application requires any special attention or additional security measures, the application proxy provides this possibility [15, 16].

2.5 VIRTUAL PRIVATE NETWORKS (VPN)

Virtual Private Networks (VPN) is a technology that combines private and public networks. An organization with multiple sites can take advantage of this technology and form a private network that uses the public network (e.g. the Internet) to send and receive data packets [13]. Therefore, the term "Virtual" is deduced from the appearance of a completely private network – when this actually is just an illusion – by the fact that the corporate networks are connected to the public network. Consequently, the confidentiality and the integrity of the data packets, sent across the unsecured public network, can be violated [17]. VPNs must therefore adopt some security mechanisms, that implements services such as confidentiality and integrity of information.

Furthermore, VPN software is implemented and configured to run on each router on each of the organizations’ sites. This configuration makes it possible for the VPN system to perform two essential tasks.

First, each router is configured to know all the IP addresses of all the other VPN routers on every site. Thus, all data packets that do not originate from and are not destined for any of these corporate VPN routers, are rejected. This configuration takes advantage of the conventional firewall’s filtering mechanism (see section 2.4).

Second, before a data packet is transmitted across the insecure public network, it must be encrypted by the VPN software. With regard to Figure 2.2 the “information channel” remains encrypted. This is...
achieved by adding security protocols such as IP Security (IPsec) onto the Internet Protocol (IP), which is
the main protocol used out on the Internet. Hence the communication is kept secure even if an opponent
manages to eavesdrop on the transmitted messages – no valuable information can be deduced from the
encrypted messages \[15\] [18]. The cryptographic techniques and algorithms used by IPsec is assumed to be
computationally secure [8].

There are several VPN technologies available on the market, but the far most attractive and most used
VPN technology is based on IPsec [15]. We will therefore, give an overview of the IPsec protocol, without
going into too many details, in the subsequent section.

2.5.1 IP Security (IPsec)

IPsec operates at the network layer (see figure 2.5.1) and does not affect the upper or lower layers – it simply,
as the name implies, provides security on IP level. Hence, no modifications to the applications are needed.
This results in that, applications that have some or no security mechanisms implemented will be ensured
with secure communication in means of network connectivity. By introducing IPsec into the establishment
of VPNs, it would be a cost-effective solution for organizations. This is due to the fact that IPsec provides
flexibility in the sense that it can be deployed in many different firewall and router devices from different
vendors. Thus, still follow the same policy rules defined for IPsec, whereby the required security mech-
anisms can be implemented for any application or network parameter (e.g. firewalls and routers). This
flexibility, by implementing IPsec in network devices, will be transparent to the client/server applications
and therefore no security related changes to the software are needed. This transparency also applies to the
end-users of the applications that unawarely depends on IPsec – the organizations do not need to train their
staff on the security aspects concerning the policy rules defined for IPsec [8] [19] [20].

How does IPsec provide flexibility and transparency? IPsec’s main task is to secure IP datagrams in
a communication by using two protocols, namely, Authentication Header (AH) and Encapsulating Secu-
ritiy Payload (ESP) [21]. A third protocol is also included in IPsec architecture, which involves the key
distribution and management problems for communication facilities, namely Internet Key Exchange (IKE)
protocol. Generally, the first two security protocols define what encryption and authentication algorithm(s)
to be used to provide the required security services. To provide these security services they must rely on a
secure exchange of cryptographic keys, which is the main purpose of the IKE protocol. This protocol can
also be seen as a negotiator for which an IPsec connection can be established through an agreement of a
common Security Association (SA). These three protocols enable the defined security services in X.800 to
provide strong security features as; user authentication, data confidentiality, and integrity. Where the users
can authenticate themselves with digital certificates or shared secret keys [8] [18] [21]. An overview of digital
certificates can be found in [8].

A security association can be best described as in [17], that is, a SA is a collection of the defined
policies and different cryptographic keys and algorithms used for a secure information exchange depending
on which IPsec operation mode (tunnel or transport mode) that is used. Furthermore, if two entities have
agreed on securing the traffic with both AH and ESP, two SAs, one for each, are needed. In other words, a
single SA cannot implement security with these two protocols simultaneously.

In order to keep track of the different SAs, the IPsec header contains a field called Security Parameter
Index (SPI) – which uniquely distinguishes a SA from another. With this SPI value, which also resides
in a communicating entity’s SA database, an easy lookup can be performed whenever a SA is used. Sub-
sequently, both the sending and receiving entity can extract the information associated with each SA, and
interpret how the information is to be processed [9]. The information which can be found in the SA database
is the SPI value, the mode, the different cryptographic algorithms with their corresponding keys, sequence
numbers and more [9] [21].

The two protocols mentioned above (AH and ESP), to secure the IP datagrams, can operate in two
modes. These modes are briefly described here below with reference to the figures 2.5.1 and 2.5.1

TRANSPORT MODE

The transport mode ensures security in means of data integrity and authentication of IP packets for protocols
above the IP protocol (e.g. transport and application protocols). This mode, according to the figure 2.5.1,
mainly protects the contents in the payload – that is the common protocols used on the public network –
TCP, UDP or ICMP packets. A point-to-point connection (e.g. host-to-host) can easily be established with
the usage of this mode. If a communication is set up from a host to a VPN gateway, one can only access
the VPN gateway directly and not further. Thus, the internal network cannot be accessed on the other site,
assuming that the two endpoint devices share a protected key to be used in the authentication process [8] [19].

With regard to figure 2.5.1(a), when AH is used in transport mode, it will only authenticate the payload
and the IP header, or selected fields of the IP header (more detailed and technical description of AH and ESP
No confidentiality measures are provided, since the payload is not encrypted. However, AH still assures that the packets’ content, while in transit, have by any means been modified. Therefore, the devices that run the VPN software can filter the traffic properly through user and application authentication when entering the premises.

![Authentication Header (AH)](image)

![Encapsulating Security Payload (ESP)](image)

**Figure 2.3:** Transport mode.

In contrast to AH, ESP provides the unsupported confidentiality service in AH. This is achieved by encrypting the contents in the IP payload (see figure 2.5.1(b)) and if an authentication service is required – this could also be provided – however, this will only authenticate the IP payload and it is optional. Thus, the IP header itself is not authenticated or encrypted, by the fact that ESP was originally designed to be used together with AH. Therefore, the passive attack like traffic analysis can be performed by an opponent on all the transmitted packets. Still, this ESP transport mode is an attractive solution for any application that needs the support of a confidentiality service since, the service does not need to be implemented in the application software. Further, since the ESP Header itself is authenticated, no replay attacks can be performed. Due to the fact that this header includes an authenticated sequence number fields.

Furthermore, "ESP auth” field in figure 2.5.1(b) is computed on ESP Header and the encrypted part of the packet. When a packet arrives at its destination, the device first checks this field, if the packets is authentic, decryption is performed, else the packet is rejected.

**Tunnel Mode**

In tunnel mode the AH and ESP works a bit differently. The general idea of the transport mode is to insert the different headers (AH or ESP) after the original IP header. While in tunnel mode (see figure 2.5.1), the idea is to keep the native IP packet unbroken and surrounding the IP packet with a new IP header and AH or ESP fields. Note that, in this mode, the IPSec headers and the original payload etc, are the payload of this new packet. As result of this tunneling mode, the intermediate network devices cannot examine the inner header fields. Hence, the hosts on the private networks, on each site, are more secure since their IP addresses are hidden by this encapsulation.

The tunnel mode is a more desirable choice when several private networks are engaging in a communication. This make sense, though tunnel mode is established between the corporate security gateways, where the IPSec is implemented. A security gateway or a VPN gateway can either be a firewall or a router. That is, a tunnel is created from the boundary of a private network on one site to another private network boundary on the other site. Thus, the individual hosts does not have to implement IPsec in order to participate in a secure communication. All the additions and strip-offs, of the new headers are performed by the diverse security gateways if needed.

It is also worth mentioned that ESP in tunnel mode, theoretically, is not as secure as AH in tunnel or transport mode, in means of authentication. Due to the fact that the new IP header in ESP is not authenticated. Yet still, the security it provides is adequate, though the inner IP header and payload is both authenticated and encrypted. This inner part is still the essential part of the new packet when processing it at the sending and the receiving security gateways.

**2.5.2 Site-to-site VPN**

Site-to-site VPNs are used whenever organizations need to connect several corporate networks destined on geographically distant places, which may also include other organizations’ and companies’ networks. In other words, an organization extends its security parameter beyond their private network (e.g. a LAN).
Hence, the IPsec policies on each site must be configured and managed properly, with well defined and common IPsec rules on each VPN device along the virtual path (tunnel) between the security gateways. Since IPsec operates in different modes and uses different protocols – multiple SAs must be commonly agreed upon. For this reason, configuration of these rules on each VPN device in the entire network (on each site) can be a very complex procedure. Therefore, if there are any conflicting rules or any misconceptions in the definition of the IPsec policies – the security parameter can be more vulnerable for potential threats and attacks, thereby inconsistent and insecure transmissions can significantly increase.

Generally, hosts on a site-to-site VPN do not run any VPN software, instead the hosts are sending packets to the VPN gateway which is responsible to process the packets. Thus, IPsec is not used within the local network, instead the packets from the hosts are ordinary IP packets which are sent to the VPN gateway where they are examined against predefined rules and routing tables. If the packets are destined within the same local network, the packets are just forwarded to the destined host without any modification. Otherwise the VPN gateway encrypts and encapsulates the entire original IP packets into new IPsec packets in order to forward them to the remote network. Now, the new IPsec packet has an outer IP packet with a new IP header, and the intermediate routers will only be able to process the outer IP header and not examine the inner IP packet’s contents. Upon receiving a packet on the gateway, the packet header is stripped of, and the packet is decrypted in order to obtain the information from the inner IP packet, allowing it to be routed through all the routers in this remote internal network – finally, it reaches its destination host. For the attentive reader, it is obvious that site-to-site VPNs employ ESP in tunnel mode at the VPN gateways (see tunnel mode in Section 2.5.1).

Furthermore, if a host sends a packet that is destined somewhere on the global internet, it passes the VPN gateway without the addition of IPsec, assuming the host requests for a particular service that is permitted by the gateway. This must however be stated in the IPsec policy.

### 2.5.3 Remote-Access

Remote-access allows traveling employees to work from home or other remote locations and still have access to an organization’s private network in a secure manner. Remote-access can be implemented with various protocols, however we will focus on IPsec. It is important to understand that although site-to-site VPN and remote-access are different technologies, they are generally used in conjunction to provide access to a corporation’s private network regardless of the geographical location of the user. In contrast to site-to-site IPsec VPNs, remote-access VPNs require VPN software to be installed and configured properly on each user’s computer. These types of software are usually referred to as VPN clients that implement IPsec. Differing from the site-to-site concept, where the computers in the private networks send ordinary IP packets to the VPN gateways, where IPsec is implemented for the secure transportation of the packets between the private networks.

More thoroughly, a remote user who wishes to communicate with the private network, establishes a tunnel with the remote VPN gateway. Whereby the VPN gateway redirects the remote access request to a remote access server (RAS). A Remote-Access Server is also called VPN concentrator, which is a special-purpose device that incorporates authentication and encryption mechanisms. This RAS should also preferably be physically secured at the main corporation network, implicitly that it is logically secured in means of secure communication with the remote user. According to Figure 3.1 the RAS is placed behind
the firewall – that is, the VPN tunnel has terminated at the router and the inner packet is forwarded to the firewall – the firewall filters the packet and sends it to the DMZ, where the RAS is located. Now the remote user can be authenticated at the RAS to be able to access the authorized resources at the main corporate network. We will discuss this further with consideration to the given topology in Section 3.1.1.

2.5.4 Demilitarized Zone (DMZ)

We have introduced the general concepts of VPNs and how IPsec can be employed into the VPN technology to offer security in means of secure communications. However, the general idea of VPN – a host user, either a remote worker or a host from the remote private network, can have access to the internal network and its valuable resources on the remote site. If commonly used services, that are provided by SMTP-servers, FTP-servers or Web-servers – if these servers reside in the internal private network, an outsider can exploit their weaknesses and gain unauthorized access to the internal network. As a consequence to this, DMZs have been designed to work in conjunction with VPNs and segregate these servers from the private internal network. Which brings us to the notion of what a DMZ is – according to [23] it is simply a small network that is comprised by these common servers (and some more) mentioned previously. DMZ could be viewed as a “neutral zone” between the private network and the public network.

All the inbound traffic is filtered at the firewall, which the DMZ is connected to. Therefore all the critical resources and organization’s valuable data is protected, and segregated from the DMZ by the filtering mechanisms of the firewall. With properly defined rules in the firewall, companies and organization can provide the services to untrusted users outside the private network, without compromising the security of the private network. This is a major reason why organization and companies today design and implement DMZ [23].
3 PROPOSED SECURITY MODELS

In this chapter we will introduce five different security models that are based on some of the general security models and technologies used to increase security in an application. Throughout this chapter, each proposed model will resemble one specific security model that will be described in general terms. The models will also elaborately be constructed (designed) with the distinctive features that are offered by that specific security model. Furthermore, some additional features will be presented as a "possible" solution to achieve the adequate level of security which is not supported by the general model. Hence, the first two questions in section 1.1 regarding the defined problem will be addressed and answered in this chapter.

Our five security models deploy security within a range, where the application is the starting point and the network perimeter is the ending point. These five models will be evaluated with respect to what kind of security services they provide, and what security mechanisms are implemented with respect to X.800 – the specific components and building blocks that are used – also the strengths and weaknesses of the various models and where they are placed in the range and why. Here below, all five security models are listed starting with the outermost:

- **VPN** – provides security on the perimeter of a network and secures every application used behind the appliances.
- **Application-Proxy** – security is still applied on the network perimeter but with a great difference – application-proxys only secure specific applications.
- **Filter** – our third model works in conjunction with the application and runs on the same computer platform as the application – providing security to the application by filtering the outgoing and incoming data.
- **Plugin** – security is applied to an application, generally by using the APIs and libraries provided by the application.
- **Built-in to the software** – we describe the pros and cons of embedding security to the original implementation of an application, with respect to the software engineering process.

We will also present several perspectives regarding the involved people in means of application security development. Generally, how they are affected by the different security deployments listed above. By placing the security on different places throughout the range, it will affect the involved people in various ways. Also, the impact on the application will be evaluated regarding the level of re-design and configuration needed, with respect to the placement of the security.

With the involved people we refer to; **developers** – that is, the people involved in the implementation of the application and aim to implement the specific services defined in X.800; **network administrators** major tasks are to maintain, update, monitor, configure and design the network infrastructure proposed by our models – and they are also more concerned with CC certified software and hardware; **security managers** are the ones that define the overall security policy of an organization – they have the responsibility to mediate and describe the security policy to all involved parties. They can also do similar tasks as the network administrator; and the **chief information officer (CIO)** has the responsibility of setting new policies, planning and coordinating management activities.

Eventually, each proposed model will serve as a design guidance and a development proposal, by which we will emphasize their strengths and weaknesses. Thereby, try to provide the best possible solution for securing an application as objectively as possible.

3.1 VPN DESIGN

In the following subsection we will propose a VPN approach with a specific structure and focus on the different design considerations involving security issues. This VPN topology’s strengths and weaknesses will be discussed and assessed with respect to possible threats and attacks an organization can be exposed to. Furthermore, the topology can be adequate for an organization while inadequate for another – depending on the security policy defined by an organization. However, our topology will provide at least some of the security services defined in X.800. Finally, the VPN model will be presented as a framework, which covers most of the architectural elements defined in X.800 to provide security. The model will also provide security against external threats and diverse attacks.

It is also worth mention that there are many VPN technologies based on different protocols, but we will only take IPsec protocol into consideration. Due to the fact that IPsec is standardized [18] – it can function on different network equipment of different manufacturers. Hence, IPsec makes it possible to be adopted into the already existing infrastructure (e.g. routers, switches, hubs, firewalls, etc) of an organization. As
mentioned previously in this thesis, IPsec is the most popular technology used in the current VPNs. Therefore, choosing IPsec by this fact, was our primary decision and also by its interoperability with different network devices and operating systems.

3.1.1 Proposed Topology

First of all, in order to understand the proposed topology in this section, we need to make some assumptions about the establishment of a VPN – both for site-to-site and remote-access VPNs. We have already mentioned that the supported technology is IPsec for securing the communications over the insecure public network (e.g. the Internet). Further, the authentication methods and what type of digital certificates (if required) to be used are agreed upon by each remote private network. Also the encryption algorithms in the IPsec architecture is predefined in the IPsec policy and the cryptographic keys are exchanged in a secure manner on every VPN gateway. Through these assumptions no conflicting rules or misconceptions of the IPsec policies of the VPN gateways are undertaken. Thereby, each VPN gateway is assumed to be properly configured.

In the proposed VPN topology (Figure 3.1) we can see that the VPN tunnels (the red, thick lines in the figure) extend from the local VPN appliance to each remote VPN appliance and not further. At the tunnel endpoints, all the traffic is encrypted and decrypted by the devices that implement IPsec. In Figure 3.1, IPsec is implemented in the VPN appliance on each corporate network. That is, the appliances in the figure are the VPN gateways that conforms the assumptions stated above.

The VPN appliances on the corporate networks implements IPsec when an IP packet is leaving the network – travels securely in the tunnels to the VPN appliance on the main corporation network – where the decryption of the IP packets are performed and the authorized resources in the main network can be accessed directly. The VPN appliance performs the key-exchange procedures required by IPsec protocol (e.g. through digital certificates) in order to authenticate the IPsec packets and to give proper, predetermined, privileged, access to the main corporation network. Note that – the VPN appliances in Figure 3.1 are not protected with any firewalls – although, we assume that they provide an adequate level of security. That is, IPsec policies are thoroughly and commonly decided upon to inhibit possible misconceptions. Also, the communication with the third party (e.g. with a Certificate Authority) for the verification of the digital signatures and secret keys are performed in a secure, trusted manner. A Certificate Authority (CA) is a trusted government or financial institution which has a directory with public key and user ID pairs that are signed by the CA. Thereby, the users can provide their public keys to the authority in order to obtain a certificate [8]. The cryptographic keys, needed for encryption, decryption and authentication are frequently updated by the VPN appliances, whereby providing further increased security.

When a remote user wants to access the main corporation network, the user must authenticate oneself
advance, which makes the adoption of certified products more desirable. Since the level of security is already tested and evaluated. Thus, the required security can be assured in that they meet a set of functional requirements which is beneficial for the organizations and companies, by Common Criteria (CC). However, the different appliances used in this topology can be certified by CC, administrators and security managers regarding IPsec.

A table of the specific services implemented by this model), which is a configuration issue for the network decisions. These services are then implemented by the security mechanisms in X.800 (see Appendix D for He/she has to make sure that all the defined security services in the security policy are covered by the design. Finally, the CIO has many design decisions to make before putting this topology into practice.

The drawbacks of this topology is that it is difficult to configure all the appliances and maintain them (e.g. troubleshooting, updating, etc). The cost of equipment is usually high since this VPN solution has specialized appliances – expertise network staff is usually needed in order to install, configure and maintain the assigned appliances, thereby enhancing the overall performance.

The obvious advantage of this topology is how the different services are managed and divided into different dedicated appliances. Since the entrance to the corporate network comes down to a single point, that point would be heavily loaded if it were to provide all the necessary services and also perform encryption/decryption, authorization and authentication checks, etc. Therefore, the information passes the entrance point and is managed only by the dedicated appliance, that is, VPN traffic is filtered by the VPN appliance, while the dedicated firewall filters all other traffic. Thus offloading the traffic by directing it to the assigned appliances, thereby enhancing the overall performance.

Although performance is not the only advantage of this topology – security is also increased – since this topology offers multilayered security, where different appliances provide different security mechanisms. Even though we still have a single-point-of-failure at the edge router, the services can still be provided, assuming the edge router is up and running. For example, if the VPN appliance has crashed, non VPN traffic can still flow in the network through the firewall and non VPN services can still be accessed. This also applies to the firewall, assuming that it is down – the VPN traffic can still enter the corporation network, but only if the traffic comes from another corporate network. Consequently, the remote users will not be able to access the corporation network, since the authentication server is destined behind the firewall, in the DMZ.

The drawbacks of this topology is that it is difficult to configure all the appliances and maintain them (e.g. troubleshooting, updating, etc). The cost of equipment is usually high since this VPN solution has specialized appliances – expertise network staff is usually needed in order to install, configure and maintain the proposed VPN topology.

When it comes to remotely accessing the main corporation network – we will always face possible threats in the presence of poor system security at the client side – where the system may have various malicious software installed and the private network will be vulnerable to a range of threats.

With respect to the application(s) used within the security perimeter, that is, the application(s) used in a VPN communication, do not need to have security implemented, though the VPN appliances offers the security. Therefore, the developers of the applications are not affected by this VPN solution in means of security implementation. Thus, this proposed solution employs security as far away from the application as possible. However, this topology requires extensive involvement of the network administrator and the security manager regarding, configuration, troubleshooting and maintenance of the appliances used in this design. Finally, the CIO has many design decisions to make before putting this topology into practice. He/she has to make sure that all the defined security services in the security policy are covered by the decisions. These services are then implemented by the security mechanisms in X.800 (see Appendix D for a table of the specific services implemented by this model), which is a configuration issue for the network administrators and security managers regarding IPSec.

By the fact that no security is implemented in the applications in the given design – it will not be certified by Common Criteria (CC). However, the different appliances used in this topology can be certified by CC, that they meet a set of functional requirements which is beneficial for the organizations and companies, since the level of security is already tested and evaluated. Thus, the required security can be assured in advance, which makes the adoption of certified products more desirable.
3.2 APPLICATION-PROXY DESIGN

This section will present the security model of the general application-proxy firewall described in Section 2.4.1. We will provide an assessment for this model by exploiting its weaknesses against active and passive attacks performed on a networking application. Then, give a possible solution on how this application-proxy firewall cope with these types of attacks. That is, which security services can be implemented by an application-proxy in order to secure different applications. This is generally performed through analyzing weaknesses of the underlying protocol used by the application in question.

The placement of the application-proxy firewall is on a dedicated server at the border of the local network, as a gateway. Whereby, all the traffic from and to the local network goes through this server. Thus, the security is applied outside the application – transparent, on the dedicated server, as a proxy-service. Therefore, no modifications on the application is required, indirectly, the developers do not have to enforce security to the application. Nor do you need to invest in a special hardware – simply install the proxy software on a server computer.

A gateway is usually placed between an internal private network and the outside public network (see Figure 3.2). The gateway is situated between these networks for the following reasons; first, there is a single contact point to the outside network, thus monitoring and logging of connections and traffic flow is made easy, which facilitates the tasks for the network administrator and the security manager. Not only can proxy-services be easily configured, new ones can be applied at a single point or existing ones can be updated, and yet securing all the applications using the service. Also troubleshooting is of interest to the managers and administrators; second, the application-proxy’s ability to do user authentication, an organization can configure it to allow only certain users to access the public network. Thereby, internal threats are reduced. By configuring the application-proxy to examine all the outbound traffic, an internal user cannot sneak out valuable information out of the private network through the proxy. Thus, organizational, internal information leakage is prevented. Through user authentication, only authorized users can gain access to the requested resources, thus implementing the access control service.

Upon receiving a request from a client application, the proxy acts as a client to the outside network, and sends a request to the destination on behalf of the client application. On a successful establishment to the destination, the proxy has two connections open, and the packets is now inspected and forwarded between the two connections by the proxy. The proxy is able to inspect packets all the way up to the application layer, which enables filtering and decision making with more information available and thus more complex filtering rules can be used. This is very useful since there are numerous ways an attacker can provide valid information in the lower layers, namely layer 3 and 4 while having malicious code in the payload. Nevertheless, software trespassing can be detected through pattern recognition or signatures for possible and known malware by examining the payload against any anomalies in various application layer protocols.

However, an application-proxy is immune against traffic analysis, in the sense that an opponent cannot gain any information about the host computers on the internal private network. They are invisible to the opponent, only the traffic between the opponent and the application-proxy can be analyzed. Because the packets contain information about the outside interface card (see Figure 3.2), like the requested data (data in the payload) and the IP address of the application-proxy firewall. Consequently, the internal private network is protected against opponents, but not the content of the transmitted data. Therefore, no confidentiality or integrity of data is provided by this design.

Note that, there is no reason to implement mechanisms against active attacks as, masquerade and modifi-
cation of messages. When an opponent masquerades as a valid user, he/she may want to damage the server offering a particular service, probably through some malicious code. This code will never pass through, provided that the proxy examines the payload against known malicious contents. Thus, anyone, a valid user or an opponent can only access the offered services only when the packets are approved by the more intelligent filtering inspections provided by this design. Moreover, a denial-of-service attack can easily be prevented by allowing 4 to 6 unsuccessful attempts, maximum, from a given source. Then, blocking packets from this specific source for a predetermined amount of time.

With respect to CC, the same reasoning as with the VPN Design in the previous section, applies also for application-proxy firewalls. That is, there are several application-proxy firewalls out on the market that are CC certified to evaluation level EAL4. An example of this is for instance Sidewinder, which can be found at http://www.securecomputing.com/index.cfm?sec=855

### 3.3 Filter

This filter model we provide here, resembles the well known network protocol Secure Shell (SSH), in the sense that all inbound and outbound traffic are exchanged over a secure channel. Since SSH is implemented on OSI layer 7, it is possible to secure all TCP traffic over an unsecured network as the Internet through application tunneling. We will therefore use the words *tunnel* and *tunneling*, when describing our model, though this model adds the security the application originally lacks.

Almost every application that uses the TCP protocol, as the underlying communication protocol, can take advantage of this model and filter the outgoing and incoming TCP traffic. That is, if TCP packets carry any application protocol, these packets are filtered before they leave the host computer. Furthermore, the filtering mechanism that this model offers, establishes a secure tunnel (as SSH) between the source and the destination (the red thick line in Figure 3.3 represents a tunnel). These filters can be implemented without any prior knowledge of the specific application protocol and are located on the same computer system as the application. Hence, any application can communicate securely through the tunnel, by the fact that the security services are applied on the TCP packets, where the application data is encapsulated.

We have implemented a prototype of this model in Java, where the security services are applied on the outgoing and incoming streams. This is our solution to not incorporate any changes into the original implementation of any application requiring security. However, a minor configuration is needed for applications that take advantage of this model, namely, the destination IP address must be configured to communicate with the filter. The filter listens for incoming streams on a specified port as a daemon process on the same platform - applies the security services on this stream - and forwards the traffic to the final destination. On the destination platform, another filter daemon must be up and running that listens for incoming traffic from another filter, hence a tunnel is established from one filter to another.

This tunnel lasts during the entire communication session. To make this tunnel secure, in means of communication, Diffie-Hellman Key Exchange algorithm is implemented. By adopting this algorithm, to exchange a secret key between two or more users in a secure manner, this key can be used as a session key. That is, a key used for encrypting and decrypting subsequent streams. The introduction of encipherment on the transmitted data provides the confidentiality service which gives a protection to passive attacks as “release of a message contents” (e.g. eavesdropping). In our implementation we also provide data integrity, implicitly, by the fact that the whole stream is encrypted.

However, there is no use to provide confidentiality or/and data integrity if no authentication mechanism is implemented to authenticate the session key exchanged with Diffie-Hellman. This is basically because the Diffie-Hellman algorithm does not provide any protection against masquerade attacks. An additional approach is required, namely, the usage of public key cryptography. This way, the session key will be authenticated between the communication entities in the following way: Host A, and Host B has exchanged their public keys in a secure manner (either in person, or through a trusted third party). Host A encrypts the session key with Host B’s public key and visa versa and an exchange is made. Both parties then decrypt the message, containing the encrypted form of the session key, with their private keys. The result is then compared with the original session key. If they match, the session key is authenticated and can be used for encrypting messages during the whole communication session.

From the involved peoples’ point of view, the adoption of this filter won’t affect the developers in any way, though this filter is already developed with the soul purpose to secure any application communicating over the TCP/IP protocol. The main task of the network administrator and security manager, is to configure and maintain the filters, which can be time-consuming since the filters are locally resided on each computer. Also, gathering and verifying the trustfulness of the public keys in the organization’s keychain, are taken out by the security manager. In fact, this might be the most important task of the security manager – no public keys can be unverified, or this model will fail in all other security services. The CIO’s job regarding the installation of the filter is to make sure that all the defined security policies, requirements, both functional
Figure 3.3: Filter.

and non-functional, are fulfilled. In other words, making sure that the filter actually implements the policies.

In the development process for this model, the security services are implemented with respect to those services defined and explained in X.800. By enforcing security mechanisms to implement these services – an adequate level of security has been achieved – even though this filter is just a simple prototype.

3.4 Plugin Design

Another way to add security to an existing application is by providing a plugin to that application, which implements the desired security services lacking or even existing. Before we go any further with how plugins can be developed for a host application, it is important to emphasize that; in order for a host application to support plugins, it must provide the developers with an Application Programming Interface (API) for interoperability to be a fact. An API is a software interface written in the same programming language as the application providing it. The API usually defines many interfaces and by implementing these interfaces, interoperability with the host application is possible. By using the host application’s API it is now possible to add the security services required and let the security mechanisms (the implementation of the security services) be implemented by the plugin.

This is however not an easy task for the developers, nor is it a straightforward process of implementing the provided interfaces. The developers have to familiarize themselves with the API; how to implement its services to achieve the desired functionality; also, from a software engineering perspective, how to use the API with respect to maintainability of the plugin. Not only is this a time-consuming procedure, it can also be a very expensive story for organizations willing to develop a security plugin, in means of staff-training hours, assuming no prior knowledge of the API exists. However, it is hard to give a real life estimate of the costs, though there are many factors involved. But training and hiring new personnel will always be an expense the organizations or companies cannot predict.

With a plugin, the security is “attached” to the host application almost as if it was implemented directly into the host application’s software. Paradoxically, the host application is not changed or configured in any way, instead it searches for installed plugins and upon encountering a plugin, the application can use the services provided by the plugin. This is due to the fact that both the host application and the plugin are using the same common API. When extending an application’s functionality in this way, the plugin does not communicate directly with another application (i.e. a networking application), instead, the application communicates with another application by using the services provided by the plugin. This way, the plugin is transparent to the communicating applications and provides an ad hoc solution for incorporating security into the applications.

When the security policies are defined by the security manager and the CIO, it is then up to the developers to tailor the plugin to implement these policies. However, the developers can only contribute with the implementation of the functional requirements stated in the policy. The non-functional requirements (reliability, scalability, and cost) described in the security policy, that is, the constraints of the specified security services are the responsibility of the security manager and the CIO. It is difficult to point out how the network manager is affected with this plugin design described in this section. But it is fair enough to say that their responsibility, regarding the management of the network, will not change, since configurations on the network infrastructure are not needed.

It is fully possible to tailor a plugin with security mechanisms to implement all the security services described in X.800. Although, the CIO must make an assessment of the cost and time to carry out this kind
of challenging development process. This design requires a tradeoff to be made between needed services and the costs of implementing a plugin, and still fulfill the adequate level of security.

In Section 3.4.1 we will give a simple example of how a plugin can be implemented.

### 3.4.1 Example

![Figure 3.4: Plugin design.](1)

This example demonstrates how a plugin in a Java-environment can be designed and implemented. We will describe all the steps and components depicted in Figure 3.4 – how they interact with each other – what they represent – and also how the execution flows.

First of all, the host application must define an API and provide it to the plugin to implement. In this case, the interface («SecurityInterface») define Java-methods, that is, how the plugin is supposed to behave and what data-types are returned, etc. In other words, the plugin implements the interface depending on the choice of the developer(s). The choices could include, encryption/decryption and other various kind of security specific properties.

Next, the host application must determine which plugins are available and also where they could be found. Upon locating the plugins, the application loads and instantiates them. Further, by writing a PluginSecurityManager one can determine what system operations an application is allowed or disallowed to perform (1). A PluginSecurityManager extends the SecurityManager class provided by the Java API, and by implementing the inherited methods, one can assure that the security policies, regarding system features, are implemented. If the plugin-code is trying to perform any illegal operation an exception is thrown and then caught by the JVM (2). The steps (3, 4 and 5) can be summarized as following; you need to implement a ClassLoader class; then call the method loadClass; finally the PluginClassLoader loads the desired plugin (e.g. the Class-object located in the plugin directory defined for the application). Thus, by employing a SecurityManager class and a ClassLoader class into the host application, the desired trustfulness of the plugin is achieved.

Now as the plugin is running, the method calls (6) are performed by the host application, that is, the implemented security services are executed. These services are implemented by the plugin and since a security manager is already defined in step (1). When errors are caused in step (6) that violated the defined system security policy, the JVM handles this error by “catching” (7) and returning it to the host application (8). Note, if the plugin does not perform any illegal operations, the plugin keeps running quietly as if it was implemented in the original software and performs its tasks as it ought to.

This example is just an overall description of how a plugin can be developed in Java, and gives an insight to the reader of the different tasks that need to be performed.

### 3.5 BUILT-IN TO THE SOFTWARE

So far we’ve only introduced security to networking applications with additional components, either as software component(s), hardware component(s) or both. In-explicitly, we have stated for our models above, that security is added after the application is already designed and implemented. There are however many people proposing that security ought to be integrated into the software development life-cycle, unified
with system engineering. They also propagate that security should be seen as a process where security is considered throughout the whole development life-cycle. This way in means of maintainability of the application, the security is acknowledged as an activity from a software engineering perspective.

During each iteration of the development process, developers together with security engineers must aim to meet the required level of security. Mainly by carefully designing and implementing the software requirement specification, but also in parallel, to implement the specified security policy, e.g. to follow the carried out security plan (see A.3). This way of developing, forces traditional process paradigms to incorporate security engineering along with the application or system design. Process paradigms are often referred to as software process models such as, The waterfall model, Evolutionary development, Component-based software engineering, etc.

Now the question arises; why do organizations avoid this approach on common basis to produce secure software that not only have very few defects, but also have high level of security? First of all, organizations must hire experienced personnel or train the existing ones, in order to adopt to the new ways of practicing software engineering. Second, the common perception among today’s businesses and organizations is that, functionality is the most important feature, whereas attributes as security and quality are often considered as afterthoughts. And finally, the efforts needed in developing secure software with high-quality, are believed to be too expensive. Furthermore, organizations and businesses are also skeptical, that changing their current process paradigms as mentioned previously, will not be beneficial regarding time, effort and costs – even though existing evidence dictate the contrary. However, they cannot neglect the inevitable initial costs when changing their development processes to produce secure software.

Adopting the secure-software design principles, by improving the current software-engineering practices, is mainly introduced to reduce the errors (defects) in software design and implementation, thereby reducing security vulnerabilities as well. Studies made by Computer Emergency Readiness Team Coordination Center (CERT/CC) [27], have shown that more than 90 percent of software security vulnerabilities are a consequence of different types of software defects. Generally, software always has errors in the code and the quantity of these errors can be estimated. A well-known term is "Kilo Lines Of Code" (KLOC) which indicates how many errors a piece of software has on average. For example, a value of 0.07 defects/KLOC indicates that there are 0.07 defects per thousand lines of code. Defects and errors in the code generally lead to vulnerabilities in the software that can be exploited. Thus, striving towards a low KLOC-value is of great importance for an organization. It is quite common that organizations today produce software with values up to 7 defects/KLOC which means thousands of defects in todays systems consisting of millions lines of code [27]. Therefore, the lower the KLOC-value – the fewer the defects – hence more secure software.

Improving or changing the traditional process paradigms affects the whole way the organizations or companies operate. For instance when integrating the Team Software Process (TSP) into the organization’s software engineering processes, the developers, managers and other staff, receive training in secure design, secure implementation, secure review and inspection and secure testing. Managers receive guidance from certified TSP coaches, that have expertise in how to increase productivity and efficiency, deliver high-quality software within budget, and more. However, the CIO is the first person that informs all the developers and other staff, that TSP shall be used as the new software engineering process for all projects. This is usually stated in a written policy by the CIO and then they receive the additional required training described above.

When the software engineers have received the proper training for TSP, the developers can implement the desired security services described in X.800, that is, using X.800 as a supplementary to the new knowledge mentioned above (e.g. secure design, implementation, etc). What TSP offers, in means of security, has nothing to do with security services or mechanisms – it is just another way of approaching software development. Evidently [27], the applications that need security will be more secure than applications developed with traditional software development approaches.

Using Common Criteria to assess applications that have security built-in to the application is possible both for open source software and commercial software. To find defects in the code, a CC evaluation assumes to have the source code available, in order to achieve EALs 5-7. However, commercial software vendors do usually not share their source code and consequently, EALs over 4 is therefore not feasible, because the higher levels require source code inspection. However, if you provide your source code for inspection, it does not mean that the code is available publicly. It is only available for the people who perform the special tests in secluded laboratories.

Using software engineering process models as TSP, shows that EALs 5-7 are often reached and adequate level of security is provided in a standardized manner. Not only does CC certified applications with EALs 5-7 assure almost defect-free software, but also, indirectly, the secure-software paradigms (e.g TSP) gives assurance that higher EALs are more likely to be accomplished.
4 Analysis

This section provides an evaluation of how the different models secure the application. In the previous chapter, we have tried to give an assessment of how an adequate level of security can be achieved within a range and with respect to the application. By providing security to applications in a network environment in various ways – ranging from securing the network perimeter, with no modifications to the application software, to building in the security into the software – we have emphasized different security issues related with each model. These will be analyzed with respect to the given problem domain in Section 1.1.

4.1 Virtual Private Networks

This model makes the application more secure by protecting it from external attacks – with aid of implementation of diverse security protocols, that already incorporates thoroughly defined defense mechanisms. With proper configurations of the different appliances depicted in Figure 3.1 a satisfactory level of security can be assured to the whole network and not only to the application(s). Assuming that no misconceptions or miss-configurations of security policies are present of course.

The obvious advantage of VPNs is that the security is applied at the network perimeter, by which every application within the internal network is secured and not visible to the external network, due to the encapsulation mechanisms offered by IPsec. This means that applications can be added without any modifications to them, or at most some minor configuration modifications in terms of IP addresses, access-lists, etc. Furthermore, our proposed VPN design uses multi-layered defenses, where each appliance offers a barrier that a possible opponent must cross. Note that, the applications themselves do not necessarily need to incorporate any security services, however, they will still be secured (in means of network security) by the security services implemented in the appliances.

Virtual Private Networks ought to be the primary security model for any larger organization or company that consists of different corporate remote sites, and wish to exchange confidential and valuable information across the untrusted public network.

4.2 Application-Proxy

Just as with VPN, an application-proxy is situated far from the application, but still in the same private network and here is where the similarities end. Application-proxies offer security to applications by addressing the various, analyzed, weaknesses and vulnerabilities in the underlying protocol used by the application in question. Thus, any vulnerability that could be exploited by internal or external attacks can be inhibited through inspection of packets all the way up to the application layer. These packet inspections also thwart the possibilities of malicious code from entering or bypassing this single point of defense. Furthermore, application-proxies that implement the access control service enables security mechanisms to only allow authorized users to gain access of resources.

When an application is configured to communicate with the application-proxy, there is no other way to contact the application besides through the application-proxy, which makes the application secure from any unwanted attacks.

Application-proxies are implemented specifically for each application as proxy services, which can be comprehended as a major drawback. This because each application needs a careful protocol examination before any security can be offered to the application, thus different implementations and designs require time, effort and money. Furthermore, one protocol does not resemble another, so reusability is not an optional choice for this model.

Finally, this model does not offer, probably the most important security services in our opinion namely, confidentiality and integrity. This is impossible in the sense that, when packets leave the network (encrypted and signed) the reverse must be performed on the other end (decrypt and verify), which is not a general deployment of an application-proxy.

4.3 Filter

This model is a perfect solution to implement the abstract descriptions of the security services described in X.800 – all outgoing and incoming TCP-traffic is secured – hence security is provided to any application in question, by simply having a daemon-process running on the same platform. According to Figure 3.3, the filters communicate with each other and not application-to-application. Thus, the specified security policies can be implemented within the filter and no changes to the application is required.

Furthermore, this daemon-process (filter) is transparent to the application, which makes it possible to reuse the design and implementation of a filter. It is also important that the communicating filters use the same byte-order if they are designated on different platforms. In other words, if one filter on a platform
uses big-endian and one filter on a another platform uses small-endian, then these filters have to rearrange the bytes transfered over a network to the same byte-order. However, in Java the endianness is performed behind the scenes by the Java Virtual Machine. With respect to this, byte-order rearrangements were not considered when we implemented our filter.

4.4 PLUGIN

Plugins are a very common way of adding functionality to applications that developers have forgotten or left out intentionally. In means of security, the application can be offered security features if the original application provides the developers with an API. As we mentioned previously, the application using (executing) the plugin is not changed what so ever. It must have been designed and implemented originally to offer an API, with the intentions to be extended in the future.

Plugin-development is a language and platform dependent task, it requires a deep knowledge and understandings of its language and offered interfaces. It is therefore a great challenge to an unexperienced developer to familiarize themselves with the environment the application is executing in. From personal experiences of developing a plugin in the Java environment – we encountered that it is not enough to know how to implement the desired functionality – you have to be familiar with the inner workings of the environment the plugin will operate in. This could for instance include: file permissions, execution flows, system security, I/O operations, etc. So there are more to developing a plugin than meets the eye.

4.5 BUILT-IN TO THE SOFTWARE

This model introduces security into the software with a unification of software engineering and security engineering in the same software development life-cycle. This approach advocates that traditional software design principles are not adequate to produce almost defect-free software. Therefore, practices to develop secure software, as for example Team Software Process (TSP), is needed. This forces organizations and companies to change their software development processes and discreetly improving them. There are evidence showing that, when trying to build security into the applications, that this approach is the best way. Simply because, using the methodology leads to low KLOC-values, and high security goes hand-in-hand with low KLOC-values. Thus, reducing errors in the implementation and the design, consequently, the vulnerabilities are also reduced.
5 DISCUSSION

This paper’s intension was to provide a structured and considered security approach to secure networking applications, both for organizations and even small businesses (10-15 employees), in order to give an insight of the different models that can be adopted when an organization needs increased security. We have taken general and well-known security solutions, and tried to propose different approaches through literature studies. And by using Recommendation X.800 as our underlying framework of discussing security, we believe that the proposed security models offer adequate security; of course only theoretically. We have interpreted each security service for each and every model, but due to the time constraints, no financial possibilities and no equipment, almost no practical work could be carried out.

However, we managed to translate some security services into Java-implementation for the filter in Section 4.3. This was mainly to get a perception of how it is to provide security as a software component for an application that was originally developed without any security services. We can easily state that implementing security by using Java’s security API was quite easy. Since we only applied security onto the TCP traffic, we didn’t need any prior knowledge of what different protocols, message syntax, etc., the application was using.

It is also worth mention, regarding CC, that the applications in their entireness do not necessarily need to be evaluated, instead only add-ons to the application as a filter, a plugin or similar need to be verified by CC.

Furthermore, it is needed to be said that the involved peoples’ tasks can vary in the sense that an employee in one organization might not have the exact same tasks as another employee in another organization, even if they have the same title. The literature on the other hand describes their tasks according to their designation, which we have conformed to. Depending on which model an organization wants to deploy, the involved people will get affected in different extents.

Although the models are quite different from each other, reasoning about the security services and mechanisms is possible by the fact of X.800’s abstract description. Each model solves security issues by implementing desired or required security services. But it is up to the organizations themselves to define how to implement them in a more technical and more detailed manner. Thus, the involved people need to possess some security expertise in order to understand how the security services should be implemented.

5.1 CONCLUSIONS

Considering what we have discussed previously in this chapter, and also assessing our findings presented in Section 3, we can draw the following conclusions that give answers to our defined problems in Section 1.1, namely:

1. We can easily conclude that our proposed models in Section 3 are all securing an application in one way or another by simply following and implementing the abstract security descriptions offered by X.800. Furthermore, the offered security abstractions allow an independent interpretation of how an application can be secured, which we have accomplished with our proposed models. In other words, these models are our interpretation of how an application can be secured. A perfect complement to this is to take advantage of Common Criteria (CC) evaluations, by which security flaws can be revealed and thereby also possibly be prevented by refining the implementation of each model.

2. Providing security to an application with an additional component can be achieved in various ways, ranging from network perimeter security to application specific security. With respect to X.800, security components can be tailored with desired security services that meets the required level of security defined in the security policy. By introducing an additional component to add security services lacking in an application in question, or even refine (maintain) the component that implements the security services offers flexibility. Consequently, application functionalities are irrelevant which facilitates for the involved people, though their primary focus are the security implementation in the component(s). These types of security components are mainly those depicted and described in Section 3.1.1 3.2, 4.3 and 5.4.

3. First of all, security is placed within the range from the network perimeter to the platform the application is operating on. The impact on the application and the involved people vary depending on where within the given range the security is applied (see Section 5). Another important distinction between the models is that, the first four models provide security by the use of additional components (2), whereas, the final model advocates for security to be considered as a software engineering process. Subsequently, not only is the networking security addressed by this model, but also the application security (e.g. data security).
5.2 Future Work and Improvements

First of all, and obviously a practical solution and verification to our findings might be of a great interest to actually present concrete data. Data, or proof that the models can actually implement the mentioned security services with desired security mechanisms. A documented proposition of what and how different technologies, equipment and cryptographic algorithms can be adopted, would be a great complement to X.800. Since both security services and mechanisms are quite abstract in X.800.

Second, we present a model of how to implement a plugin in Section 3.4.1 in Java, which could easily be translated to an actual Java-implementation. However, we leave that implementation for those who are more interested in the practical real-life approach.

Finally, further improvements to the filter-implementation can be introduced, in means of additional security services and mechanisms. It is worth mention that a more fashionable and common way to implement the exchange of the secret keys, would be through a trusted third party. However, the most important improvement would probably be to implement the filter as a daemon process listening on specified application(s), while being completely transparent.
REFERENCES


A SECURITY ANALYSIS

Any kind of company must realize the level of security that a company must attain in order to protect themselves from possible threats. Investigating the threat-level, security weaknesses and the vulnerabilities in a company is of great importance. To understand the potential threats that the company may be exposed to, forces the company to evaluate what assets are more valuable than other and therefore an elaborate risk analysis must be taken out for the underlying system. Physical, logical and human or organizational threats must be identified in order to take precautions against any violations of the system’s security. Gathering as much information out of the company as possible, enables the company to design a good plan of how different security related issues can be solved. As a result of the analysis, the vulnerabilities of the assets and resources can be minimized [10][7][6].

A.1 RISK ANALYSIS

During risk analysis, information of the potential risks to the organizations assets is gathered. The information that is interesting is what kind of threats that are probable of affecting the organization and where these threats may exist. The absolute first thing to do when identifying risks is to evaluate which assets that are important and needs to be protected. Assets simply means any kind of thing or item that is useful or valuable. An organization can have several different assets such as the ones classified in [29] as following:

- **Physical** – buildings, automobiles and other non-computer equipment
- **Data** – different databases, inventory records, etc
- **Software** – applications, operating systems, security software
- **Hardware** – computers, networking equipment, etc
- **Personnel** – employees, customers, vendors, etc

It is important to realize that different assets have different value to an organization. Each asset must be evaluated to see the level of contribution it provides to the organization, thus its importance must be recorded and documented. This must mainly be done because it is usually not cost-efficient to protect every single asset of an organization, inevitably a compromise must be made. When evaluating the assets, one must think of how important an asset is to the organization, and questions like ”How much is the cost to replace it?” or ”How difficult is it to replace a certain asset?” and ”How is the organization affected if a certain asset would become unavailable?” must be answered in order to categorize the most important assets which provide great value to an organization [29].

After the work of analyzing the assets is carried out, different weaknesses and vulnerabilities will most definitely be discovered. If there is not any certainty that a threat exists, an examination of the different weaknesses in the organization should still be made. That is, people must think of the current manner of how data is being stored and where; which network protocols that are currently being used; how encryption is used and over which communication channels is the data encrypted [10].

Upon discovering weaknesses and vulnerabilities in the organization, the existence of threats is obvious. The combination of threats and weaknesses makes it easy to understand how vulnerable an organizations is. Having this information in possession enables the organization to plan and design different solutions to cope with the situation.

When all the assets have been documented and categorized by their importance to an organization, the process of discovering threats (see Definition A.1) to these assets takes place. In this process, it is important to understand why threats may occur and how they can be carried out against these assets.

**Definition A.1 (Threat).** “A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability” [12].

Threats come in many forms and they can be placed in a number of categories depending on what assets they affect, who or what lies behind the threat and where the threat originates from. One type of threats is physical threats – these threats are typically stealth, fire, flood, vandalism, etc. As the name implies, these threats physically damage the assets, but we will not discuss these further in this thesis.

Another type of threats are logical threats – which is generally unauthorized use, manipulation, denial of service, etc. These are mainly considered in terms of software like, operating systems, databases, etc.
Human and organizational threats are another type of threats where the lack of knowledge, competence and also mistakes by employees can pose threats. There are numerous reasons where these kind of threats can occur. Poorly defined policies may lead to employees having poorly understood what is allowed and what is not [10].

A.2 Security Policy

The development of security policies are strongly influenced by the process of risk analysis discussed in the previous section. Inevitably, the security policies reflect and emphasize what parts and assets of an organization that is significantly more valuable and important than others [7]. The security policy does not only concern assets and resources but also employees, following from the fact that the policies are derived from the process of risk analysis where different threats are identified.

A security policy deals with the idea of having certain rules about how different equipment and services should be used and explicitly stating what is allowed and what is not allowed. Someone need to be responsible for the policies being followed by every employee. The amount of security policies in organizations differ, where some organizations and companies have security policies for every possible asset or procedure, while others have poorly defined policies and maybe only a few of them.

Security policies may be applied to how an organizations mail system is used, where it is clearly defined what is allowed and what is not. How the mail system should be used and for what purpose it should be used. How different employees working on different departments should have access to data shared among the company or maybe only parts of the company or subsections of the department. All these types of rules stated in the policies must be followed with security in mind in order to reach the security level that is preferred [10, 30].

The meaning of policies is rather vague but if a more precise explanation is needed, one can think of a policy as a suggestion of something that is of great importance but without actually giving a description of how the solution should be acquired. Thus, a security policy about how employees might log into a system, may state that the employees should log in with a seven letter password or more, but not actually state what technology that is to be used. Or even better and more general policy is to simply state that the employees must use an authentication mechanism when logging in, with nothing said about specific details. This way, the security policies does not have to change to often, and yet they still serve as a guidance towards developing procedures which follow the security policies.

A security policy defines the goals of an organizations safety precautions. This with respect to assets, threats, data, information and availability to name a few. The policy also defines who is responsible to follow which policies. Furthermore, a constant evaluation of the process of realizing the policies is done which includes what the personnel have done to fulfill the policies [7].

A.3 Security Plan

A security plan is produced based on the process of designing the appropriate security policies. The goal with the security plan is to describe how the security policies can be enforced. With a plan on how to achieve the security policies, the process of securing assets and data in an organization is straight forward. The security plan can be thought of as a road map to achieving the security policies a company has issued.

The security plan is constantly changed and updated while re-evaluation of the plan is taking place. At each evaluation it is clear what has been done and what still is not fulfilled. The security plan can also be changed upon the discovery of new security policies which are developed and need to be applied to an organization.

When the security plan has been followed correctly, the assets in the organization which previously were vulnerable by the possible threats, should have a higher level of security [7]. But in order to correctly follow the security plan, it must be more detailed technical solutions described, and that is where the security architecture comes in, described in Section 2.2.
B  RADIUS

Remote Authentication Dial In User Service (RADIUS) is an AAA (authentication, authorization and accounting) protocol for applications who need to authenticate themselves over a network. In this thesis, these applications are referred to as VPN clients. Remote users, who need to access the internal private network, send an authentication request to the VPN concentrator (e.g. the VPN appliance in Figure 3.1), which in turn communicates with a RADIUS server. Note that, with this type of configuration the VPN appliance acts as the RADIUS client, therefore the RADIUS server is transparent to the remote user, and should be placed in a protected DMZ as in our proposed topology in Figure 3.1 [31].

Furthermore, the request made by the remote user, contains a username and a password, which has the VPN appliance as its final destination address. When the "Access-Request" packet arrives at the VPN appliance, the VPN appliance hashes the username and the password using MD5 hashing and shared secret, and sends the user ID and the password encrypted to the chosen RADIUS server for verification. The reason with sending the password encrypted between the VPN appliance and the authentication server is to protect it from any opponent from getting hold of the password in cleartext while eavesdropping on that connection. If the request is successfully verified – the remote user will have authorized access to the private network. However, before notifying the remote users that they are granted access to the services offered by the private network, the authentication server has to find the username in the database that holds a list of information of which service(s), port(s) and more, the user is allowed to access. The RADIUS server prepares an "Access-Accept" response, which holds a list of configuration values that state what type of services the remote users are allowed to access.

The remote clients that are configured to use RADIUS to authenticate themselves, are usually presented with a custom login prompt where they type in their username and password. This information is then placed in an "Access-Request" packet which is automatically created.

For more technical and further description of the packet format and type, the communication session with the VPN appliance, RADIUS server and the security considerations can be found in [24].
C Pervasive Security Mechanisms

These pervasive security mechanisms shown in Table C.1 are not implemented by any particular security services. They still need to be considered in means of security management.

**Pervasive Security Mechanisms**

<table>
<thead>
<tr>
<th>Trusted Functionality</th>
<th>Security mechanisms which are accessed by different functionality should be trusted.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Label</td>
<td>Resources can be labeled with security labels which are used to describe how sensitive the data is while in transit.</td>
</tr>
<tr>
<td>Event Detection</td>
<td>Security violations that are detected can be reported or logged. And also different recovery actions can be performed.</td>
</tr>
<tr>
<td>Security Audit Trail</td>
<td>Breaches in security can be detected and investigated by several security audit trails (e.g. system records and activities are reviewed and examined independently).</td>
</tr>
<tr>
<td>Security Recovery</td>
<td>Handles requests from other mechanisms and performs different recovery tasks (e.g. event handling and management functions).</td>
</tr>
</tbody>
</table>

**Table C.1: Pervasive Security Mechanisms**
## D  Provided Security Services

This table gives an overview of what security services the proposed models implement.

<table>
<thead>
<tr>
<th>Security Services</th>
<th>VPN</th>
<th>Application-Proxy</th>
<th>Filter</th>
<th>Plugin</th>
<th>Built-In</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Access Control</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Data Confidentiality</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Data Integrity</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Non-Repudiation</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Availability</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table D.1: Provided Security Services

