UNIFYING THE CONCEPTUAL LEVELS OF NETWORK SECURITY THROUGH THE USE OF PATTERNS

by

Ajoy Kumar

A Dissertation Submitted to the Faculty of

the College of Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

Florida Atlantic University

Boca Raton, FL

May 2014

UNIFYING THE CONCEPTUAL LEVELS OF NETWORK SECURITY

THROUGH THE USE OF PATTERNS

by

Ajoy Kumar

This dissertation was prepared under the direction of the candidate's dissertation advisor, Dr. Eduardo B. Fernandez, Department of Computer & Electrical Engineering and Computer Science, and has been approved by the members of his supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

SUPERVISORY COMMITTEE:

Eduardo B. Fernandez, Ph.D., P.E.
Dissertation Advisor

Maria M. Larrondo-Petrie, Ph.D.

Mihaela Cardei, Ph.D.

Michael Van Hilst, Ph.D.

Nurgun Erdol, Ph.D.
Chair, Department of Computer and Electrical Engineering and Computer Science

PP Mohammad Ilyas, Ph.D., P.E.
Dean, College of Engineering and Computer Science

Deborah L.Floyd, Ed.D.
Interim Dean,Graduate College.

3/18/14
Date

ACKNOWLEDGEMENTS

I would never have been able to finish my dissertation without the guidance of my advisor, all my committee members, help from my family and friends, and support from my work family and my wife.

I would like to express my deepest gratitude to my advisor, Dr. Eduardo B. Fernandez, for his excellent and patient guidance all these years. The door to his office and email has always been open whenever I ran into a trouble spot in my research or had a question about my research or writing.

I would also like to express my sincere gratitude to my committee members Dr. Maria Petrie, Dr. Mihaela Cardei, Dr.Michael Van Hilst, and the members of the Secure Systems Research Group for all the valuable advice and comments on this dissertation.

I would also like to thank my parents and my sister and all family members who have encouraged me to reach this goal. I would like to thank all my friends at work. They have been my second family and have unconditionally supported and encouraged me with their best wishes.

I would like to express my heartfelt gratitude to my wife. She has been always there cheering me up and standing by my side through this entire process.

Finally I would like to thank the Beloved Almighty for the wisdom and perseverance bestowed on me during this research.

ABSTRACT

Author:	Ajoy Kumar
Title:	Unifying the Conceptual Levels of Network Security through the Use of Patterns
Institution:	Florida Atlantic University
Dissertation Advisor:	Dr. Eduardo B. Fernandez
Degree:	Doctor of Philosophy
Year:	2014

Network architectures are described by the International Standard for Organization (ISO), which contains seven layers. The internet uses four of these layers, of which three are of interest to us. These layers are Internet Protocol (IP) or Network Layer, Transport Layer and Application Layer. We need to protect against attacks that may come through any of these layers.

In the world of network security, systems are plagued by various attacks, internal and external, and could result in Denial of Service (DoS) and/or other damaging effects. Such attacks and loss of service can be devastating for the users of the system. The implementation of security devices such as Firewalls and Intrusion Detection Systems (IDS), the protection of network traffic with Virtual Private Networks (VPNs), and the use of secure protocols for the layers are important to enhance the security at each of these layers.

We have done a survey of the existing network security patterns and we have written the missing patterns. We have developed security patterns for abstract IDS, Behavior–based IDS and Rule-based IDS and as well as for Internet Protocol Security (IPSec) and Transport Layer Security (TLS) protocols. We have also identified the need for a VPN pattern and have developed security patterns for abstract VPN, an IPSec VPN and a TLS VPN. We also evaluated these patterns with respect to some aspects to simplify their application by system designers.

We have tried to unify the security of the network layers using security patterns by tying in security patterns for network transmission, network protocols and network boundary devices.

DEDICATION

I dedicate this work at the Lotus feet of Bhagawan Sri Sathya Sai Baba who is my GOD and my spiritual guide and to my aunt M.S Rugmini who is looking at me with pride from heaven. She was always interested in my work and had always wanted me to complete my PhD.

UNIFYING THE CONCEPTUAL LEVELS OF NETWORK SECURITY THROUGH THE USE OF PATTERNS

TABLES

FIGURES

1 INTRODUCTION

Information security is a growing need for organizations and individuals; which implies that computer systems must be protected against attacks [Fer01]. In the case of computers connected in a local network, attacks may come from hosts in external networks or in other local sub-networks. Network traffic has a layered structure and we need to protect against attacks that may come through any layer.

Networks are structured into layers where control is passed from one layer to the next, starting at the application layer in one station, and proceeding to the bottom layer, over the channel to the next station and back up the hierarchy. Network architectures are described by the ISO standard which contains seven layers [For04]. The internet uses four of these layers, of which three are of interest to us. These layers are:

- Internet Protocol (IP) or Network Layer - Responsible for the delivery of individual packets from the source host to the destination host.
- Transport Layer – Responsible for the delivery of a message from one process to another.
- Application Layer - Responsible for providing Internet services to the user.

Among the several issues that need to be addressed in these layers, security is one of the most critical issues and a systematic approach is required by the developers to

build and implement a secure network structure. Each of these layers is subjected to security threats and we need to consider security defenses for these layers. Security threats are detected and, stopped or mitigated, based on security policies which in turn lead to the development of security mechanisms that we have described using patterns.

The primary objectives of security are to provide confidentiality, integrity, availability, and accountability for the data being communicated. Data conveyed by messages over the network is usually vulnerable to attacks and can be targeted by many people for political, personal or profit reasons. Security countermeasures are usually classified into five groups: identification and authentication, authorization, logging and auditing, information hiding, and intrusion detection.

In the world of network security, everyday, systems are plagued by various attacks internal and external and could result in Denial of Service (DoS) and/or other damaging effects. Such attacks and loss of service can be devastating for the users of the system. The implementation of security devices such as Firewalls and Intrusion Detection Systems (IDS), the protection of network traffic with Virtual Private Networks (VPNs), and the use of secure protocols for the layers are considered to enhance the security at each of these layers.

Figure 1.1 gives an overview of the security mechanisms in the network layers according to the protocols used in each of these layers. In this figure the lines represent the different network layers under consideration and the ovals represent the security

mechanisms applied to these layers and indicate that these mechanisms can be applied at these layers in different forms.

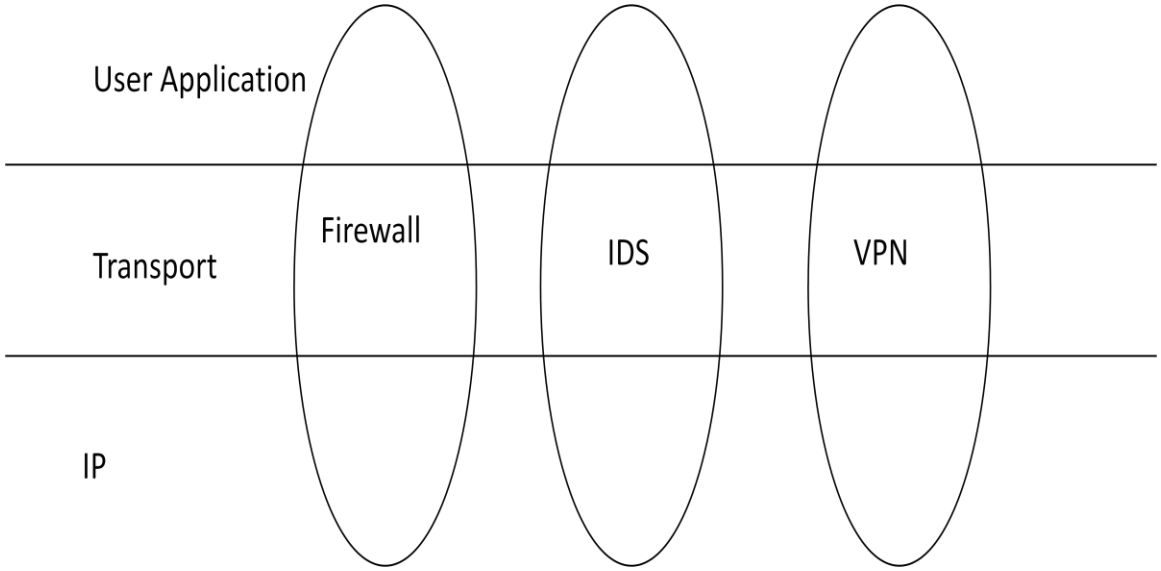


Figure 1.1 Overview of network layers and their security mechanisms.

The Internet Protocol Suite, also referred to as Transmission Control Protocol (TCP)/Internet Protocol (IP), defines a reference model for networks that includes four layers [Sta03]: Application, Transport, Internet, and Link. One can apply security to any of these layers. Two secure protocols are commonly used in networks:

- The *Internet Protocol Security* (*IPSec)*, which provides cryptographic functions at the Internet (IP) layer [For04, Sta03].

- The *Transport Layer Security (TLS) protocol*, which provides similar functions at the Transport (TCP) layer [For04, Sta03]. This protocol is based on the Secure Sockets Layer (SSL) protocol.

One can also apply security defenses at the network boundaries, where networks enter the computational nodes. Two security mechanisms are normally used in network boundaries:

- *Firewalls*, which filter input and output traffic according to predefined rules.
- *Intrusion Detection Systems (IDS)*, which try to detect attacks in real time.

The *Virtual Private Network (VPN)* is a third type of mechanism that is frequently used in the security context. Virtual Private Networks make use of public network resources to access the internal nodes of an enterprise. The transmission is protected through a cryptographic tunnel that provides message confidentiality and integrity. Since this network exists only in a virtual sense, it has been termed a virtual private network.

Figure 1.2 shows the layers and the security protocols used at each of the layers. The Application layer has different protocols based on the type of application. The Transport layer uses TLS as the security protocol while the IP layer uses IPSec as the security protocol. Application protocols such as Hypertext Transfer Protocol (HTTP), Lightweight Directory Access Protocol (LDAP) and Simple Object Access Protocol (SOAP) need to use the lower layers to support typical application tasks such as displaying web pages or running email services; they use their own version of security

protocols such as HTTPS (HTTP Secure), LDAPS (LDAP Secure) and WSS (Web Service Security) respectively.

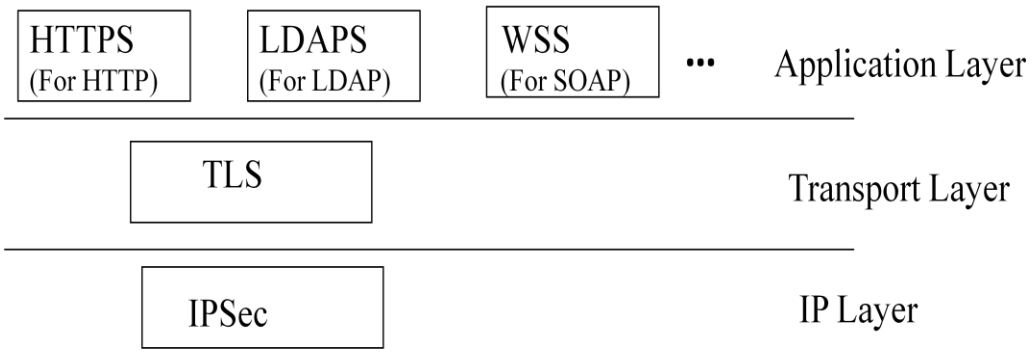


Figure 1.2 Network Layers and Security Protocols.

In this work, we make the following contributions:

1. We go over secure network design, ensuring that we have the proper choke points to enable control of traffic. We look into security devices such as firewalls and intrusion detection systems, the protection of our network traffic with virtual private networks (VPNs), and the use of secure protocols in Chapter 3.
2. We do a reconnaissance survey (Chapter 3) of available security patterns on network transmission, the boundary security devices and the associated security protocols. In the same chapter we identify the patterns that are missing which forms the basis of the patterns developed in Chapters 4, 5 and 6.

3. We also discuss briefly in our survey (Chapter 3) network patterns for specialized networks such as VoIP networks and wireless networks.
4. We develop an abstract pattern for IDS (Chapter 4).
5. From the abstract pattern we derive two types of concrete IDS patterns: (Chapter 4):
    - Anomaly or Behavior Based IDS.
    - Rule or Signature Based IDS.
6. We present an abstract pattern for network layer protocols (Chapter 5).
7. From this abstract pattern we derive patterns for IPSec protocol used at the network layer and the TLS protocol used at the transport layer (Chapter 5).
8. In Chapter 6, we describe the abstract VPN pattern.
9. The abstract VPN pattern is made concrete to define patterns for TLS VPN and IPSec VPN.
10. Based on three sets of defined criteria we do a qualitative analysis of the patterns we developed as part of this dissertation (Chapter 7).
11. The major contribution of this dissertation is that, we unify the security of the network layers using security patterns by tying in security patterns for network transmission, network protocols and network boundary devices.

This thesis includes the following chapters: Chapter 2 presents a background on security patterns and how they have evolved over time. Chapter 3 is a reconnaissance survey on the already existing and available security patterns on network transmission, the boundary security devices and the associated security protocols. In the same chapter

then we identify the patterns that were missing. We complete the picture of patterns in secure network transmission over the network layers by developing these missing patterns presented in the next three chapters. In chapter 4, we present pattern diagrams for Intrusion Detection Systems (IDS) starting with an Abstract IDS and expand the pattern into Anomaly Based IDS and Signature Based IDS. Chapter 5 presents a pattern for the security protocols again starting with the Abstract Security Protocol pattern and expanding the pattern to IPSec protocol used in the network layer and TLS protocol used in the transport layer (TLS). Chapter 6 describes patterns developed for IPSec and TLS Virtual Private Networks (VPN) once again starting with a security pattern for an Abstract VPN. We investigate the qualitative features of our security patterns, by evaluating each pattern based on three sets of unique criteria in Chapter 7. Finally Chapter 8 presents conclusions and future work.

2 BACKGROUND ON SECURITY PATTERNS

A way to understand and build systems that can counter the threats, faced by these network layers, is the use of patterns. Patterns are solutions to recurrent problems in given contexts. Security patterns have been looked at extensively in the current world of threats and have been studied in detail.

Secure systems need to be built in a systematic way in which security is an integral part of the software life cycle [Fer04]. Patterns are encapsulated solutions to recurrent system problems and define a way to express requirements and solutions concisely, as well as providing a communication vocabulary for designers [Bus96, Gam95]. From an architectural point of view, patterns encapsulate design strategies, as well as decisions or constraints, which determine a family of satisfying architectures, and result in actual (partial) architectures upon instantiation.

To design a secure system we need to evaluate the security threats to the system. A security pattern describes a solution to the problem of controlling a set of specific threats through some security mechanism, defined in a given context [Sch06]. Security patterns allow application developers to use security measures without being experts on security. We can also use patterns to evaluate existing systems by examining them to see if they contain the required patterns [Fer13].

Security patterns have gained significant momentum since their inception in [Yod97] seminal work, although similar approaches - object models of security in the context of object-oriented databases [Fer93] - appeared before that time as well. In 1998, two more patterns appeared: a pattern for cryptography [Bra00] and a pattern on access control [Das98]. These patterns opened the floodgates and a lot of patterns followed. [Sch06] was the first to try to categorize and unify a variety of security patterns. Security patterns are now accepted by many companies. Microsoft, Sun and IBM have books, papers and web pages on this subject.

There are several ways of looking at a security pattern:

- *As an Architectural Pattern:* Security patterns can be considered a type of architectural pattern because they usually describe global software architecture concepts.
- *As a Design Pattern:* Some groups consider security patterns as design patterns as security is an aspect of a software subsystem.
- *As an Analysis Pattern:* Security constraints should be defined at the highest level; that is, at the conceptual model of the application or at the analysis stage of an application.
- *As a special type of Pattern:* To create a security pattern we can add or remove sections from the standard template or use different notations and a different model making it a special pattern. (Special is defined as anything that is not standard.)

Security patterns can be applied in two ways: individually or in groups to help (partially) secure a system; or as a part of an overall, coherent approach, i.e. a pattern-based security methodology. In either case, any factors such as software distribution should be taken into account both by the patterns and by any methodology applying those patterns.

2.1 Documenting Patterns

The solution offered by a security pattern needs to resolve a set of forces and can be expressed as a UML class, sequence, state and activity diagrams, although we do not need all these representations in most cases. A set of consequences indicate how well the forces were satisfied, or in our context how well the attacks were handled. Many formats are available for documenting patterns. But for this thesis we adopt the format developed for the Pattern Oriented Software Architecture series (POSA1) [Bus96]. And the format is qualified by the following attributes:

Name:

Name and a brief summary of the pattern

Also Known as:

Other names of the pattern, if any.

Example:

A real world example demonstrating the existence of a problem and the need for the pattern.

Context:

The situation in which the pattern applies and pre-conditions for its application.

Problem:

The problem the pattern addresses and the associated forces. The forces define constraints on the solution.

Solution:

The fundamental solution underlying the pattern. The solution has two or more parts:

> *Structure:*
>
> A detailed specification of the structural aspects of the pattern using UML class diagrams.
>
> *Dynamics:*
>
> Typical scenarios describing the behavior of the pattern and indicating the messages passed between the participating objects or Actors.

Implementation:

Guidelines for implementing the pattern.

Example Resolved:

Discussion of the aspects resolved in our example problem.

Variants:

A brief description of specializations of the pattern. It could lead to a complete new pattern.

Known Uses:

Examples of the use of the pattern in existing systems [Sch06].

2.2 Use of Security patterns

The most common uses for security patterns are as follows:

- To help application developers who are not security experts to add security in their designs.
- A set of abstract security patterns can be a good description of the security requirements of a system.
- As guidelines for designers of security mechanisms to define the objectives or intended features of their products.
- To evaluate the security of existing systems. This is the most frequent application of security patterns in practice.
- To make complex standards understandable is another use of security patterns.

- To teach security concepts.
- To build prototypes or simulations.

Security patterns in practice and designing secure architectures using software are illustrated with around 70 security patterns in [Fer13].

3 SURVEY OF SECURITY PATTERNS FOR THE NETWORK LAYERS

In this chapter our goal is to try to do a reconnaissance survey on the already existing and available security patterns on network transmission, the boundary security devices and the associated security protocols. Then we identify the patterns that are missing and complete the picture of patterns by developing these patterns in secure network transmission over the network layers.

Our survey begins with considering patterns for security contexts. We look at patterns addressing issues in network functions such as identity management, authentication, secure communications and filtering as well as patterns for VPNs. This is followed by security patterns in the network layers that are associated with the network devices that include boundary devices such as Firewalls and IDS. And finally we look at and investigate security patterns associated with security protocols in these network layers. We also look at specialized network patterns such as VoIP networks and wireless networks.

3.1 Network Transmission Patterns

To protect against eavesdropping and unauthorized information disclosure a range of cryptographic techniques can be used, ensuring message secrecy, integrity and authenticity which are the major concerns of a secure channel of communication. Unsolicited network information should be checked and/or filtered, requiring the design of components concerned with data filtering. The system should be protected by components encapsulating authorization and access control, but these components are not normally network components.

3.1.1 Cryptographic Algorithms

In this section we briefly provide some (relatively superficial) background in cryptography. There are three kinds of cryptographic algorithms commonly used to make sure the message secrecy and integrity [Uzu13].

3.1.1.1 Encryption ciphers

Encryption ciphers are algorithms that use a key (series of bits) to transform or convert plain text to unreadable text called the cipher text in the same alphabet. Ciphers can be symmetric or asymmetric. Symmetric ciphers use the same key for both encryption and decryption. Asymmetric ciphers use a key pair – a “public” key known to all parties, and a “private” key known to one party only. Symmetric algorithms can be further broken down according to whether they operate on blocks of data or data streams.

3.1.1.2 Cryptographic hash functions

Cryptographic hash functions reduce input data of some (arbitrary) length to an output of fixed length in a one-way, collision-resistant fashion. Hash functions can be categorized requiring a cryptographic key for operation or un-keyed, and within these categories can be further divided according to their purpose as Modification Detection Codes (MDCs) or Message Authentication Codes (MACs), respectively. Both MACs and MDCs can be built from symmetric cipher algorithms, e.g. DES with a pre-generated key.

3.1.2 Digital signature schemes

A digital signature scheme is the asymmetric-key analogue of a MAC, with the additional feature that digital signatures provide non-repudiation. In a digital signature scheme a public-private key pair is generated, the private key (still a series of bits, that can be interpreted as a series of compound values) is used to sign the message via a signature function, and the public key is used to verify the signature using a verification function [Uzu13].

3.1.3 Authentication Patterns

Authentication in network devices is used to verify the identities of users, processes or nodes and to protect against identity/name spoofing. Authentication uses specialized protocols and requires a more careful design when done over multiple nodes on a network.

Brown et al. [Bro99] present an Authenticator pattern for application level, distributed object-based systems by using the Gang-of-Four Abstract Factory pattern [Gam95] to create authenticated objects. Hays et al. [Hay00] in whose work the Authenticator pattern is used as part of a pattern-based security framework for distributed object-based applications, suggest that the Strategy pattern [Gam95] could be used for varying the authentication algorithms in the Authenticator.

Fernandez and Warrier [Fer03a] present a Remote Authenticator/ Authorizer pattern, for network-based authentication and authorization. The pattern is based on forwarding a user's authentication request to a centralized proxy (authentication) server, which can then delegate the request to a server located somewhere else.

Two entities can mutually authenticate by two separate instances of the Authenticator pattern, supported by a certification infrastructure as in the mutual authentication pattern of [Lu08].

3.1.4 Secure Channel Pattern

Secure communication channels must be set up between any communicating entities in a network, using internet to communicate, to protect against eavesdropping and unauthorized information disclosure. A range of cryptographic techniques can be used for this purpose, ensuring message secrecy, integrity, and authenticity depending on the needs of the system.

A number of patterns also address cryptographic concerns for communications in particular. [Leh01] for example, present a pattern language for key management within

the context of secure communications by identifying the existence of three main key management stages during secure communication: (1) key generation, (2) key exchange and (3) key storage.

[Bra98] defines Tropyc: a pattern language for cryptographic software. Tropyc's aim is to cover the security mechanisms of message communications by providing patterns for (1) secrecy (via encryption), (2) integrity (via message digests or a Modification Detection Code), (3) authenticity (via a Message Authenticity Code) and (4) non-repudiation (via digital signatures). [De01] describe a Secure Client/Server Communication pattern in an attempt to capture the design of secure session establishment in the context of a client/server system, along with a set of cryptographic patterns supporting the necessary key management.

The Secure Client/Server Communication pattern proposes the addition of a security layer [Yod97] at both ends of the communication, containing a single secure socket component at each end to handle the cryptographic functionality. The stages of establishing the session include (1) (symmetric) session key generation, (2) session key distribution/exchange using asymmetric encryption (with RSA) and (3) usage of the session key to encrypt subsequent communications. One salient feature of the pattern lies in the generation and use of two session keys (one for each communicating end) as opposed to the usual use of one symmetric key [Fer10b] in secure communications.

[Has09] takes a more detailed approach and present a set of patterns for symmetric and asymmetric encryption, XML encryption and digital signatures, largely

within the context of web-services, but applicable to a more general range of systems as well. Symmetric Encryption [Has09], in particular, extends and specializes the secrecy pattern, simultaneously placing greater emphasis on design strategy. The digital Signature pattern in [Has09], is a significant improvement over the closely related Message Integrity (MDC-based) pattern in [Bra98] dealing with the same concerns, notwithstanding the merits possessed by the latter.

Together with the Secure Channel patterns, the cryptography patterns presented in the above references form a closed set of patterns for secure communication in a distributed setting [Uzu12].

Figure 3.1 shows the conceptual model for the Secure Channel pattern [Uzu13]. This model shows how to achieve the security properties such as integrity, authentication, secrecy and availability of a secure message channel established between a client and a server communicating messages over a channel.

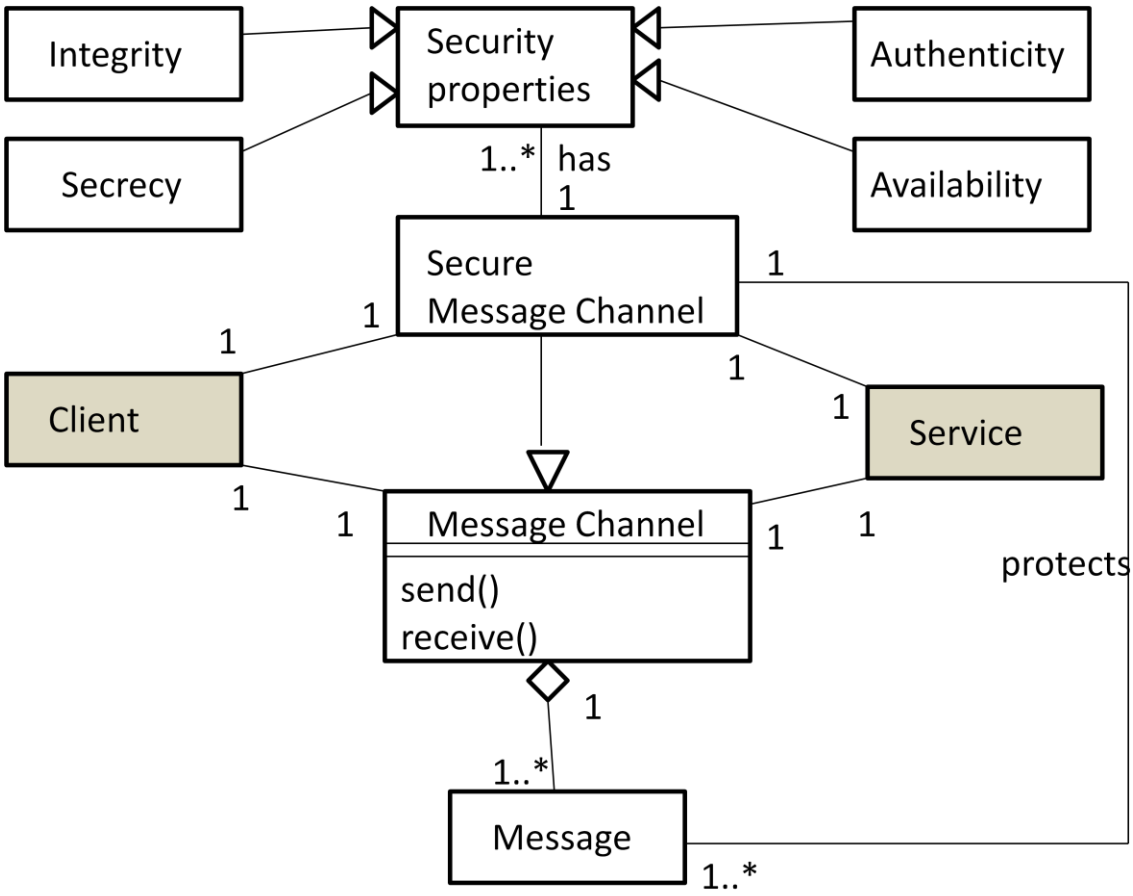


Figure 3.1 Conceptual model for the Secure Channel Pattern [from Uzu13].

3.1.5 Filtering

Unsolicited network information should be checked and/or filtered in network communication requiring the design of components concerned with data filtering. [Fla99] describes the Data Filter Architecture pattern, whose intent, as the name implies, is to filter unwanted information from a data stream communicated over a network according to a set of pre-defined policies.

3.2 Boundary Patterns

3.2.1 Firewall Patterns

[Del04] and [Fer04] present Application and XML Firewall patterns, which are essentially composite patterns consisting of Firewall [Fer03b], Authorization (particularly Role Based Access Control (RBAC}) [Fer01] and more generally what is referred to as Policy Based Access Control (PBAC) [Del07], Authentication [Fer03] and Reference Monitor [Fer02]. Most of these patterns are catalogued in [Fer13]. The purpose of the Application Firewall is thus not so much to filter as to authorize (network) requests via a policy enforcement point. These enforcement points can be deployed in either a system-wide centralized location or on a per application basis. In the latter case the Application Firewall is an application of the Single Point of Access pattern [Yod97]. In both cases authentication information is stored in a centralized storage area for easier management. Network requests or accesses are intercepted by the firewall's policy enforcement point(s), authenticated, and authorized by checking their permissions against the access control rule set. The Application Firewall pattern can, therefore, be seen as an application extension of the corresponding OS authorization pattern functionality [Sch06]. The XML Firewall is an Application Firewall tailored mainly for web-services, adding XML specific content inspection features.

3.2.2 IDS Patterns

A system intrusion is any attempt to attack a system and compromise its security aspects such as integrity, confidentiality, or availability. Intrusion Detection Systems

(IDS) are implemented to detect an intrusion when it occurs and on detection they should trigger appropriate recovery measures [Bie01]. IDSs monitor all traffic as it passes through a network, analyze it, reconstruct sessions and detect predefined patterns of attack or abnormal behaviors that could be caused by system attacks.

The Abstract IDS pattern defines the basic features of any IDS. An abstract security pattern defines only fundamental, implementation-independent functions and threats [Fer08]. Concrete patterns add functionalities and threats and take into account the characteristics of their specific concrete environment. In this case, the abstract functions are realized by concrete IDS which operate based on known attack signatures or based on abnormal behavior or anomaly in the network, i.e, Signature-Based IDS or Behavior-Based IDS. We identified that security patterns were missing for IDS and hence developed a pattern for abstract IDS that defines their general features and concrete patterns for Signature-Based IDS and Behavior-Based IDS in Chapter 4. The abstract pattern for IDS is shown in Figure 3.2. IDS detects an attack on a message between the server and the client using an event processor that processes the event and the attack detector detects attacks based on attack information usually stored in a database and creates a response. Details of this pattern are given in Chapter 4.

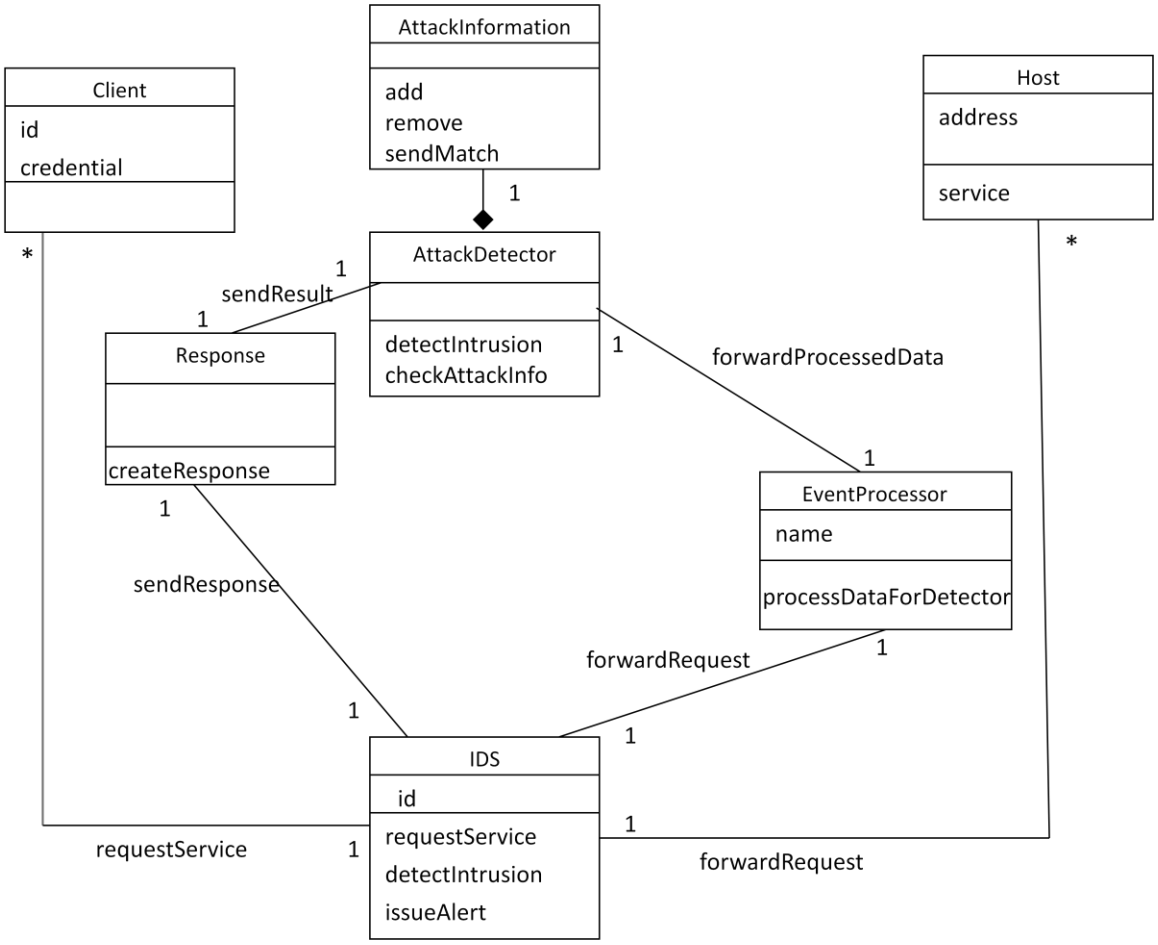


Figure 3.2 Class Diagram for Abstract Pattern for Network IDS.

3.3 Protocol Patterns

IP layer and the Transport layer are the two significant layers, besides the application layer, through which communication flows over the internet. The Internet Protocol Suite, also referred to as TCP/IP, defines a reference model for networks that includes four layers [Sta03]: Application, Transport, Internet, and Link. One can apply security to any of these layers.

IPSec and TLS are two protocols applied at the network layer and transport layer respectively. Figure 3.3 illustrates an abstract pattern for TLS and IPSec protocols. Detailed patterns are provided in Chapter 5. This pattern shows a server node talking to a client node using the protocol controller. The protocol controller uses the Authenticator and Secure Channel Patterns.

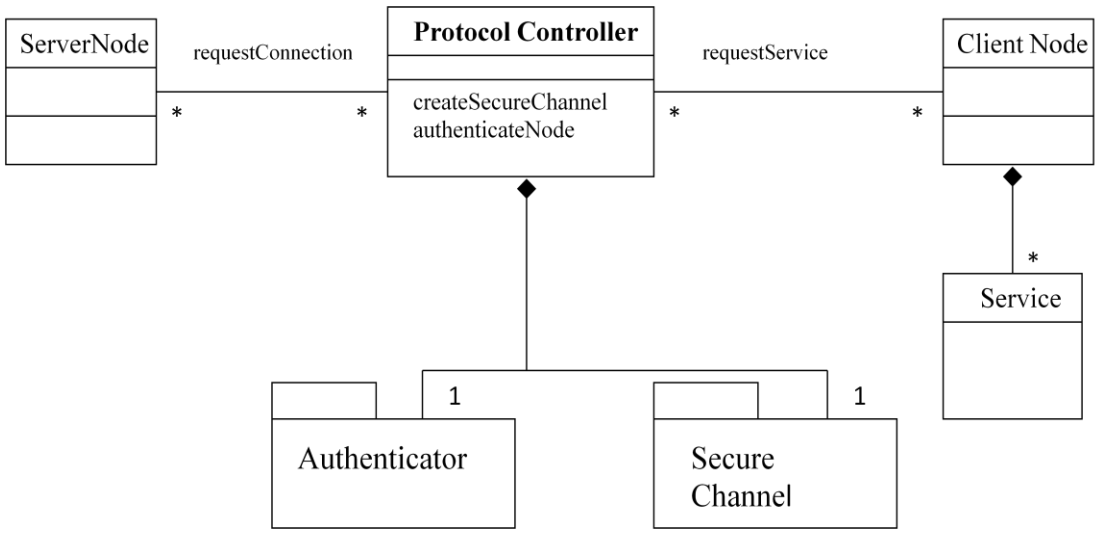


Figure 3.3 Abstract Pattern for Network Layer Protocols.

3.3.1 IPSec Patterns

Internet Protocol Security (IPSec) is a protocol for securing communications providing message secrecy and end to end authentication. The IPSec protocol, provides cryptographic functions at the Internet (IP) layer [For04].

3.3.2 TLS Protocol

Transport Layer Security (TLS) is a connection-oriented protocol that provides a secure channel between a client and a server at the transport layer of the network. TLS protocol, like IPSec at the network layer, provides similar cryptographic functions at the Transport (TCP) layer [For04]. This protocol is based on the Secure Sockets Layer (SSL) protocol.

We have subsequently developed security patterns that describe the IPSec protocol and TLS protocol, the details of which are provided in Chapter 5.

3.4 VPN Patterns

Virtual Private Networks (VPNs) provide a way for secure transmission of information over public networks. A VPN establishes a secure tunnel to send/receive information between nodes. Their use has consistently increased in the last few years and there exist several varieties of them that operate at different layers of the network architecture. We realized the need for a security pattern for an abstract VPN architecture from which concrete varieties can de derived, in particular, an IPSec and a TLS (SSL) VPN. These VPN patterns can be seen as a technological realization of the secure channel patterns mentioned above. Hence we created a set of patterns which include an abstract version together with the two technology dependent variants based on IPsec and TLS/SSL for network and transport level security respectively. Figure 3.4 shows an

abstract pattern diagram for VPN. Detailed patterns for the abstract pattern and the concrete patterns are provided in Chapter 6.

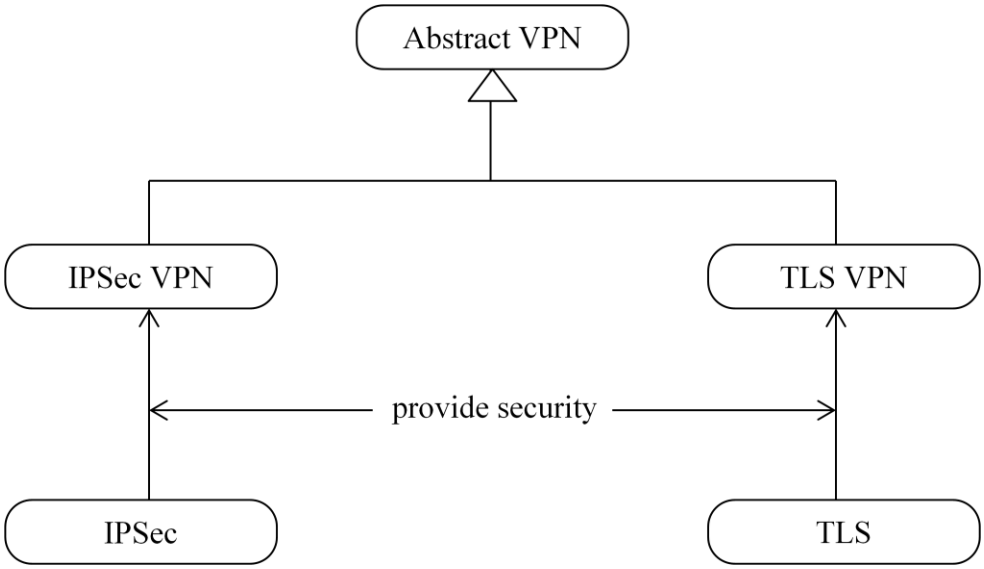


Figure 3.4 Abstract Pattern diagram for VPN patterns.

3.5 Specialized Network Patterns

A brief survey has been done on security patterns in some of the specialized networks such as VoIP networks and wireless networks.

VoIP networks:

Voice over IP (VoIP) is defined as the transport of voice as packets over IP based networks. Therefore, VoIP can be achieved on any data network that uses IP, such as the

Internet, intranets, and Local Area Networks (LAN), where digitized voice packets are transmitted over the IP network.

[Fer07] has discussed existing VoIP architectures and provided UML models for them. The approach provides a precise framework, that defines, where to apply security. This paper considers possible security attacks and relates them to the ways the system is used. It also considers some defense mechanisms and patterns for them are provided. [Pel07] presents a set of misuse patterns for VoIP: Denial of Service (DoS), Call Interception, and Theft of Service on VoIP.

Wireless Networks:

Web services security standards are used for the secure design of the communications between a web service and a mobile client and for the storage of the web service and its data. However, because those standards are designed to be flexible, they are also complex and verbose. On their part, wireless devices have specific technological constraints as well as their own standards. To use web services standards in mobile devices, it is necessary to adapt these standards to consider the limitations of portable devices. [Fer12] has shown the use of patterns as a way to adapt web services security standards to the wireless environment. Web services standards are typically long documents, e.g., the XACML 3.0 Core Specification is 150 pages long, written to be comprehensive but not easy to understand, and using a combination of XML, UML, and natural language.

3.6 Survey Conclusions

We have done a brief survey on the already existing patterns in network transmission, network boundary devices and also the network protocols used in the Internet layer and the Transport layer. We also identified the missing patterns from this discussion. Hence we have developed these missing patterns which form the basis of the next three chapters. The IDS patterns are illustrated in Chapter 4. The protocol patterns are detailed in Chapter 5 and finally the VPN patterns are provided in Chapter 6.

A qualitative comparison for the patterns developed by us, based on three distinct sets of criteria, is done in Chapter 7.

4 SECURITY PATTERNS FOR INTRUSION DETECTION SYSTEMS

4.1 Introduction

A system intrusion is any attempt to attack a system and compromise its security aspects such as integrity, confidentiality, or availability. Intrusion Detection Systems (IDS) are implemented to detect an intrusion when it occurs and on detection they should trigger appropriate recovery measures [Bie01]. IDSs monitor all traffic as it passes through a network, analyze it, reconstruct sessions and detect predefined patterns of attack or abnormal behaviors that could be caused by system attacks.

The Abstract IDS pattern defines the basic features of any IDS. An abstract pattern defines only fundamental, implementation-independent functions and threats [Fer08]. Concrete patterns add functionalities and threats and take into account the characteristics of their specific concrete environment. In this case, the abstract functions are realized by concrete IDS which operate based on known attack signatures or based on abnormal behavior or anomaly in the network, i.e, Signature-Based IDS or Behavior-Based IDS. We present here patterns for all these three types of IDS.

Section 4.2 describes an Abstract IDS pattern, defining the common features and threats of all IDSs. Sections 4.3 and 4.4 describe concrete IDS for the Signature-Based IDS and Behavior-Based IDS respectively. For the concrete patterns, we show only their

differences with respect to the abstract pattern, the assumption being that all their other aspects are inherited from the abstract pattern. Our patterns are intended for system designers, who can use these differences to select the type of pattern they need.

4.2 Abstract IDS

Intent

Monitor all traffic as it passes through a network and analyze it to detect possible attacks and trigger an appropriate response.

Example

Our company has a firewall to control traffic from the Internet. However we are still plagued by viruses and other attacks that penetrate the firewall. These attacks could be already existing attacks or they could be new attacks. We need to improve our defense against such attacks.

Context

Nodes for local system that need to communicate with each other using the Internet or any other insecure network.

Problem

An attacker may try to infiltrate our system through the Internet and misuse our information by reading or modifying it. We need to know when an attack is happening and take appropriate response.

The solution to this problem is affected by the following **forces:**

- *Communication:* The system is usually more secure if we have a closed network. However in today's world it is better and realistic to use the Internet or other insecure network to reduce costs, which may subject our network to security threats.
- *Real time behavior:* Attacks should be detected before the attack completes the purpose of attack so that we can preserve our assets and save time and money. It is difficult to detect an attack when it is happening. But such detection is imperative so we can react timely and appropriately.
- *Incomplete security:* Security measures such as encryption, authentication, etc, may not protect all our systems because they do not cover all possible attacks.
- *Non-Suspicious users:* Protecting our system through a firewall is quick and easy. However request coming from a non-suspicious address (permitted by a firewall) could still be harmful and should be further monitored.
- *Flexibility:* Hard coding the type of attack can be done easily. But it will be hard and time consuming to adapt to attack patterns that keep changing constantly.

Solution

Each request to access the network is analyzed to check whether it conforms to the definition of an attack. If we detect an attack an alert is raised and some countermeasures may be taken.

Figure 4.1 shows the typical placement of IDS in a network complementing a firewall. The firewall filters requests for services and the IDS further checks for

suspicious patterns in request sequences. If a suspicious pattern is detected, the network operator is alerted and the firewall may block some or all traffic.

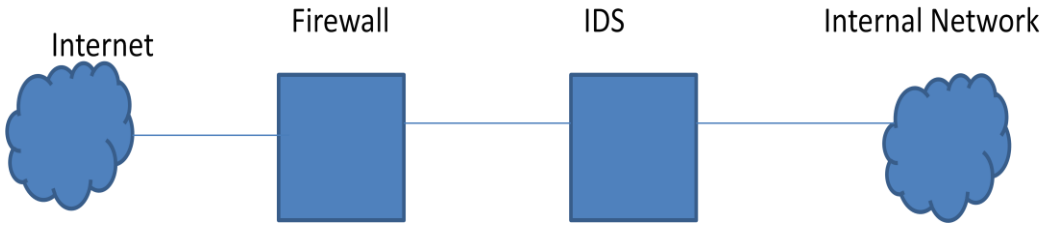


Figure 4.1 Possible Placement of Network IDS to complement a Firewall.

Structure

In Figure 4.2, a **Client** requests some service from the **Server**. The **IDS** intercepts this request and sends it to an **Event Processor**. The **Event Processor** processes the event so that the **Attack Detector** can analyze the event and implement some method of detection using some information from **Attack Information**. When an attack is detected a **Response** is created.

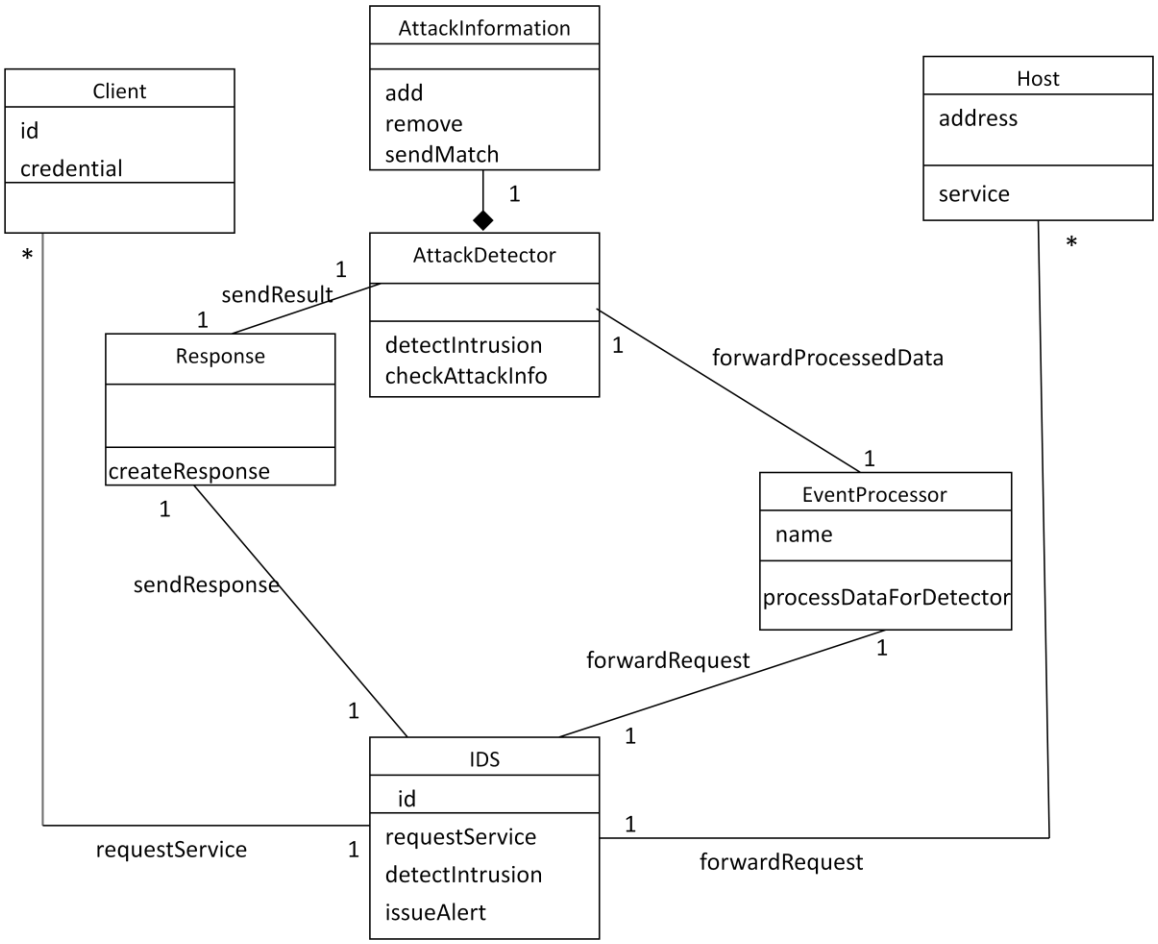


Figure 4.2 Class diagram for the Abstract IDS security pattern

Dynamics

We describe the dynamic aspects of the Abstract IDS Pattern using a sequence diagram for the following use case:

Detect an intrusion (Figure 4.3):

Summary: The client requests a service from the host. The IDS intercepts the message and checks whether the request is an attack or not and raises a response.

Actors: Client and Server

Precondition: We have attack information available.

Description:

a) A Client makes a service request to the host.

b) The IDS sends the request event to an Event Processor.

c) The Event Processor processes the event data so that the Attack Detector can interpret the event.

d) The Attack Detector tries to detect whether this request is an attack or not by comparing with the available information in the Attack Information.

e) If an attack is detected a Response is created.

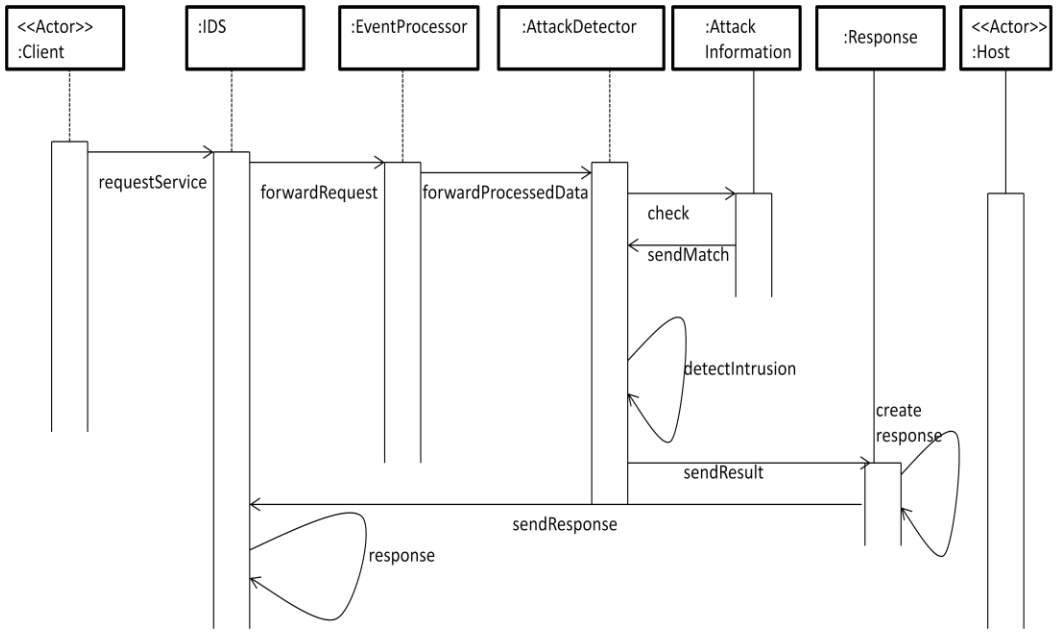


Figure 4.3 Sequence diagram for detecting an intrusion with Abstract IDS.

Alternate Flows:

1) The Attack Information may not be able to detect an attack (a false negative).

2) The Attack Information may indicate an attack when no attack is present (a false positive).

3) If no attack is detected, the request for service is forwarded to the host.

Postcondition: If an attack is detected, suitable preventive measures may be applied.

Implementation

We need to create a database with attack information so we can check against this database and decide if an attack is happening. The incoming event is compared against

the database and a decision is made whether the incoming event is an attack or not. The concrete versions of this pattern use different types of information to detect attacks.

The CIDF (Common Intrusion Detection Framework), a working group created by DARPA in 1998 mainly oriented towards creating a common framework in the IDS field. CIDF defined a general IDS architecture based on the consideration of four types of functional modules as shown in Figure 4.4 [Gar09].

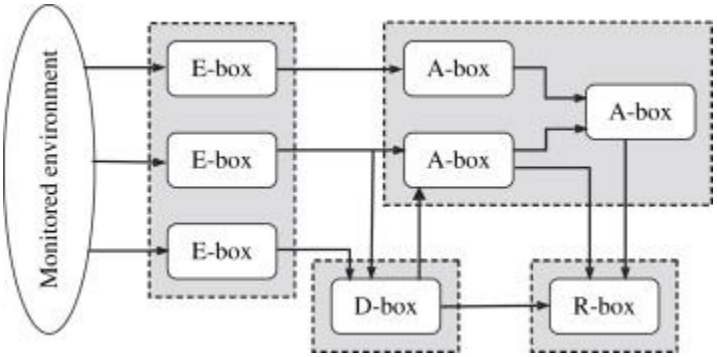


Figure 4.4 General CIDF architecture for IDS systems [from Gar09].

- *E blocks* ("Event-boxes"): This block is composed of sensor elements that monitor the target system, thus acquiring information events to be analyzed by other blocks.
- *D blocks* ("Database-boxes"): These are elements intended to store information from E blocks for subsequent processing by A and R boxes.
- *A blocks* ("Analysis-boxes"): Processing modules for analyzing events and detecting potential hostile behavior.

- *R blocks* (“Response-boxes”): The main function of this type of block is the execution, if any intrusion occurs, of a response to thwart the detected menace.

Concrete patterns

- IDS can be either signature (rule) based or can be based on anomalies (abnormal behavior). There are significant differences in their use and effectiveness. The patterns for both the *Signature-Based* IDS and *Behavior-Based* IDS are described below in the subsequent sections.
- A *hybrid model* of both the signature based and behavior based IDS together is now available. A Behavior-Based IDS detects the anomalies in traffic and then compares the anomalies with an attack signature in a Signature-based IDS.
- According to the resources they monitor, IDS systems are divided into two categories: *Host based IDS* systems and *Network Based IDS* systems. Host based IDS systems are installed locally on host machines. Host based IDS systems evaluate the activities and access to key servers within the host. The network based IDS systems inspect the packets passing through the network [Dep05]. This classification is not discussed and is out of scope for this work.

Known Uses

NID is a freely-available hybrid intrusion detection package that can be installed on a machine. NID monitors network traffic and scans for the presence of known attack signatures, as well as deviations from normal network behavior [Gra00].

Consequences

This pattern has the following advantages:

- *Communication:* If we can detect most attacks, we can safely use the Internet or other insecure networks to access other systems.
- *Real time behavior:* Attacks can be detected when the attack happens and the system can be alerted which saves the system both time and money from recovery measures and may prevent misuse of our assets. These attacks can be detected in real time if they have sufficient and appropriate information.
- *Incomplete Security:* This will be an added layer of security in addition to encryption, authentication, etc..
- *Non-Suspicious users:* A request coming from a non-suspicious address (permitted by a firewall) is further inspected and analyzed.
- *Flexibility:* The detection information can be modified to consider new attacks or new behavior.

This pattern has the following disadvantages:

- Some attacks may be so fast that it may be hard to recognize them in real time.
- Attack patterns are closely tied to a given environment (operating system, hardware architecture etc…) and cannot be applied easily to other systems. This means we need to define detection information tailored for an environment.
- There is some overhead in the addition of IDS to a system.

Related Patterns

- Firewalls can be added to complement the IDS [Sch06]. Firewalls usually deny requests made by unknown addresses. They can protect against attacks coming from distrusted sources and can block the addresses from where an attack originates.
- The response class could be implemented as a Strategy pattern [Gam94].

4.3 Signature-Based IDS

Intent

Check every request for access to the network against a set of existing attack signatures in order to detect possible attacks and trigger an appropriate response. **AKA:** Rule Based IDS, Knowledge-Based IDS.

Example

Our company has a firewall to control traffic from the Internet. However we are still plagued by viruses and other attacks that penetrate the firewall. We need to improve our defense against such attacks.

Context

Distributed systems executing applications that may provide services to remote nodes. Access to the network can be from the Internet or from other external networks.

Problem

Whenever data is accessed from the distrusted networks, there is always a possibility that this access can be harmful to the local node. We need to detect possible

attacks while they are occurring. Security techniques such as authentication and firewalls are usually implemented to provide security, but we need additional defenses to detect whether an access request is a possible attack or not. The solution to this problem is affected by the following forces:

- *Known Attacks:* It is easier to protect the system against known attacks. Many attacks are new instances of known attacks and have a well defined attack signature.
- *Completeness:* If we have a complete collection of known attacks and their signatures, it is easier to detect an attack exhibiting one of these signatures.
- *Flexibility:* Hard coding the type of attack can be done easily. But it will be hard and time consuming to adapt to attack patterns that keep changing constantly.

Solution

Detect the occurrence of attacks by matching the current attack signature against the signature of previously known attacks.

Structure

Figure 4.5 represents the class diagram of this pattern. The IDS intercepts an access request for a service. An **Event Processor** processes the information and feeds this processed information to a **Attack Detector** that tries to match the sequence of requests to the signatures in the **Attack Signature Information** and decides whether the request is an intrusion or not. If an attack is detected by getting a match of signatures, some appropriate **Response** is raised.

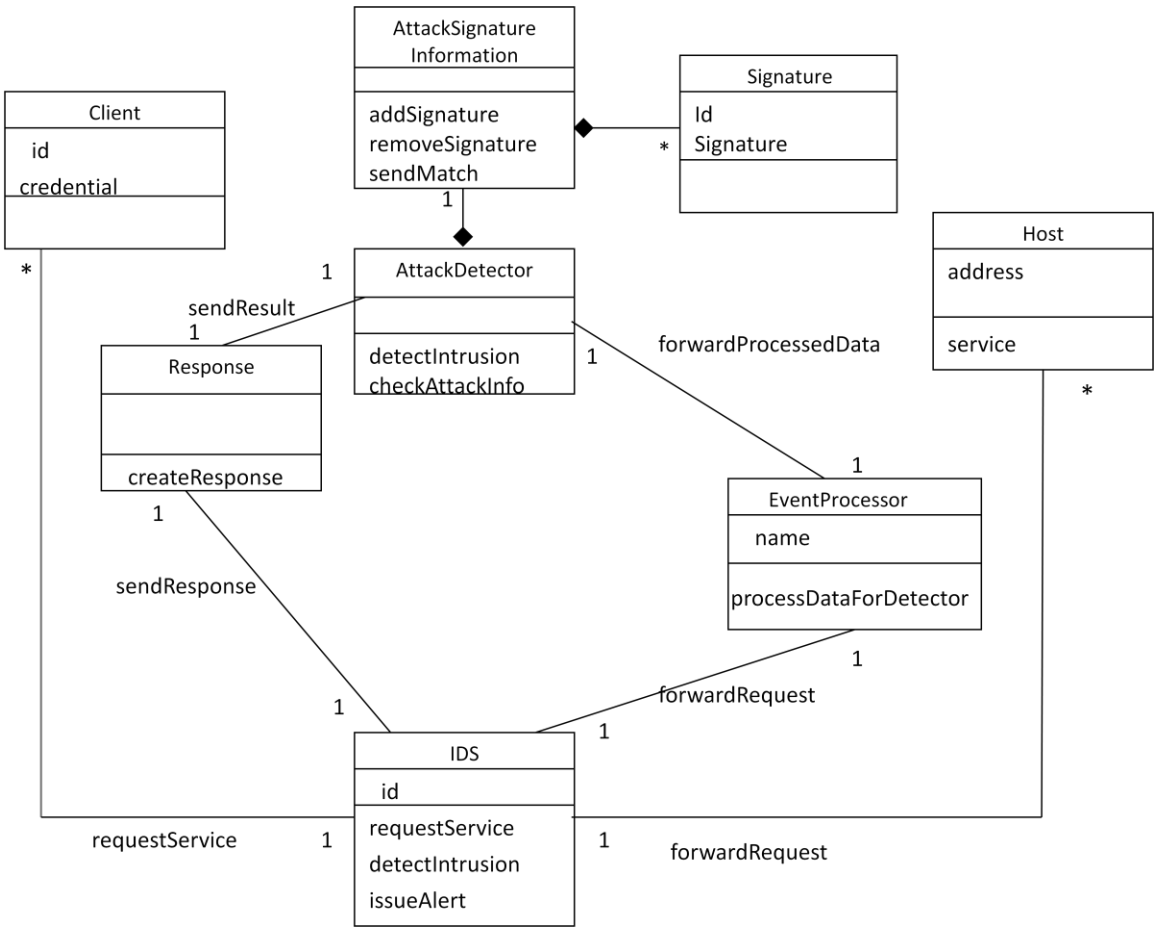


Figure 4.5 Class Diagram for Security Pattern for Signature-based IDS.

Dynamics

We describe the dynamic aspects of the Signature-Based IDS Pattern using a sequence diagram for the following use case:

Detect an intrusion (Figure 4.6):

Summary: The client requests a service from the host. The Signature-Based IDS intercepts the message and determines whether the signature of the event matches an existing attack signature and if the request is an attack, appropriate response is raised.

Actors: Client and Server.

Precondition: We have information about attack signatures available.

Description:

a) A Client makes a service request for a service to the host.

b) The IDS sends the request event to an Event Processor.

c) The Event Processor processes the event as required by the Attack Detector and passes the processed event data to the Attack Detector.

d) The Attack Detector tries to detect whether this request is an attack or not by comparing the signature of the event with the available signatures in the Attack Signature Information.

e) If a match is detected a Response is created.

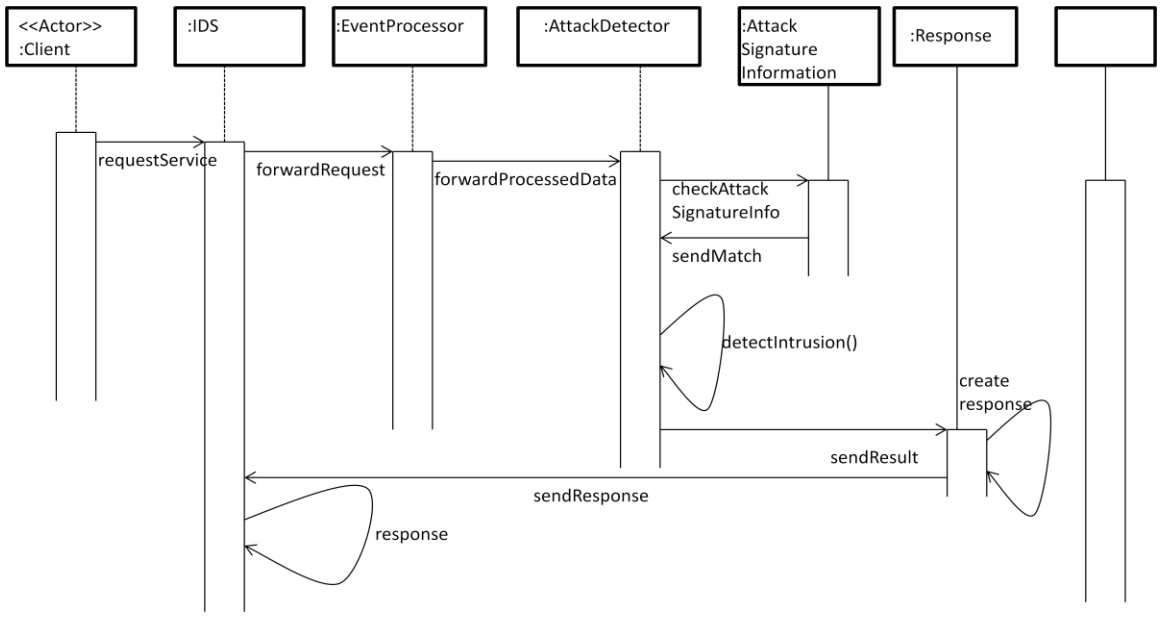


Figure 4.6 Sequence Diagram for detecting an intrusion with Signature-based IDS.

Alternate Flows:

1) The Attack Signature Information may not be able to detect an attack (a false negative).

2) The Attack Signatures can match and may indicate an attack when no attack is present (a false positive).

3) If no attack is detected, the request for service is forwarded to the host.

Postcondition: If an attack is detected while happening, suitable preventive measures can be adopted.

Example Resolved

We added an Intrusion Detection System besides the existing firewall to the system. Now any request authorized by the firewall is checked against the attack signatures to detect whether the access request is a possible attack. If we detect an attack, an alert can be raised and the firewall can block the request.

Implementation

We first need to create a database with a set of all the known or expected attack patterns. We then select a detection algorithm. Some possible detection algorithms are:

- Expression matching: The simplest form of misuse detection involves searching the event stream for known attack pattern expressions [Ver02].
- State transition analysis: The whole process is a network of states and transitions. Every observed event is applied to finite state machine instances (each representing an attack scenario), possibly causing transitions [Ver02].

- Dedicated languages: Some IDS implementations describe intrusion signatures using specialized languages varying from compiled expressions to programming languages such as Java. A signature takes the form of a specialized program, with raw events as input. Any input triggering a filtering program, or input that matches internal alert conditions, is recognized as an attack [Ver02].
- Genetic algorithms: A genetic algorithm is used to search for the combination of known attacks (expressed as a binary vector, each element indicating the presence of a particular attack) that best matches the observed event stream [Ver02].

Known Uses

- An IDS can be combined with a firewall as done in Nokia's network systems [Nok01].
- Cisco IDS utilizes detection techniques including stateful pattern recognition, protocol parsing, heuristic detection, and anomaly detection [Cis].
- LIDS is a signature-based intrusion detection/defense system for the Linux kernel [Lid].
- RealSecure [Rs] by Internet Security Systems is IDS adapted by IBM for intrusion detection packages on the market. It can monitor TCP, UDP and ICMP traffic and, if a match is found, counter-measures can be implemented along with read/write server locking, IP blocking and other measures. This product is bundled with CheckPoint Software's Firewall [Che].

Consequences

This pattern has the following advantages:

- *Known Attacks:* Detection can be effective against known attacks.
- *Completeness:* If all known attack signatures are available in the database, attacks can be detected in real time.
- *Flexibility:* It is relatively easy to add new attacks to the detection set.

This pattern has the following liabilities.

- It only works for known attacks. A new attack will not be detected. We have to constantly update the database with new attack signatures.
- Some attacks don't have well defined signatures and the attacker may disguise the signatures. This may lead to false positives and false negatives.
- Some attacks may be so fast that it may be hard to recognize them in real time.
- Attack patterns are closely tied to a given environment (operating system, hardware architecture, etc…) and cannot be applied easily to other systems.

Related Patterns

- This pattern is a special (concrete) case of the Reference Monitor [Fer01].
- The patterns for firewalls in [Sch06] complement this pattern.
- The response class could be implemented as a Strategy pattern [Gam94].

4.4 Behavior-Based IDS

Intent

Check every request for access against patterns of network traffic in order to detect possible deviations from normal behavior (anomaly) which may indicate an attack, and trigger appropriate responses. **AKA:** Anomaly-Based IDS.

Example

A company uses a public network for its applications. The network is exposed to security threats, especially a variety of unknown attacks. Their business could be in jeopardy if their customers realize the fact that their system is not secure enough.

Context

Any network application, where the temporal behavior of network traffic is repetitive and predictable.

Problem

Whenever data is accessed from the Internet or other external networks, there is always a possibility that this access can be harmful to the network. We need to detect possible attacks while they are occurring.

The solution to this problem is affected by the following **forces**:

- *New Attacks:* In today's world, the networks are constantly bombarded with new attacks that do not have a specific attack signature. We need to detect these kinds of attacks.

- *Real-Time:* We need to detect attacks in real time while they are happening, and not after the attack has happened and it is too late to recover from the attack.
- *Increased Vulnerability:* Some networks, e.g. mobile networks are more vulnerable to unknown attacks because of their mobile nature.

Solution

Observe the traffic over a network and try to find deviations from normal or expected behavior. Any deviation from normal behavior is treated as a sign of intrusion.

Structure

Figure 4.7 represents the class diagram of this pattern. A **Client** requests some service from the system. The **IDS** intercepts this request and sends it to an **Event Processor**. The **Event Processor** processes the event data as needed by the **Attack Detector** and passes the processed data to the **Attack Detector** which involves a process of establishing profiles of normal behavior which can be compared with the current load in **Behavior Profile Information**. When an attack is detected a **Response** is created.

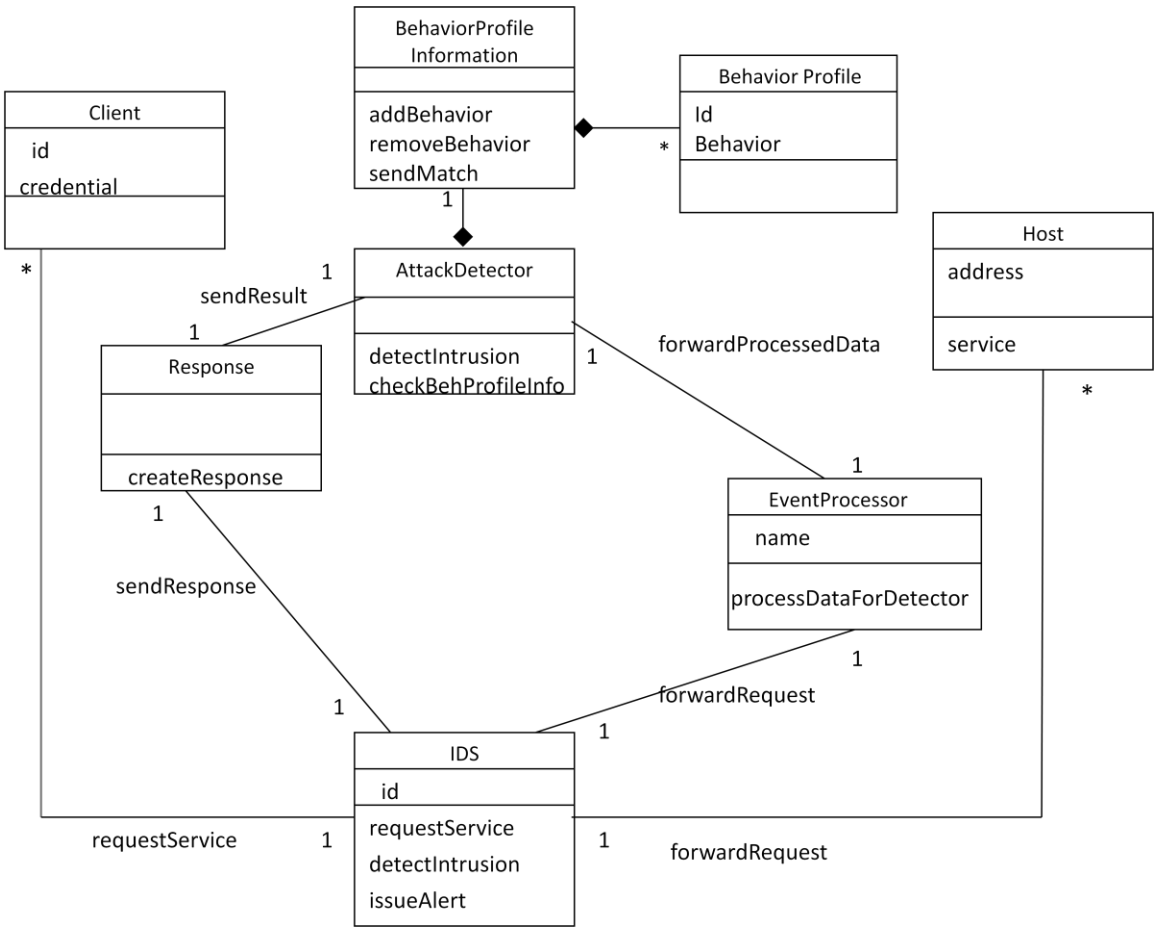


Figure 4.7 Class Diagram for Security Pattern for Behavior-based IDS.

Dynamics

We present here the dynamic aspects of the Behavior-Based IDS Pattern using sequence diagrams for the following use case:

Detect an intrusion (Figure 4.8):

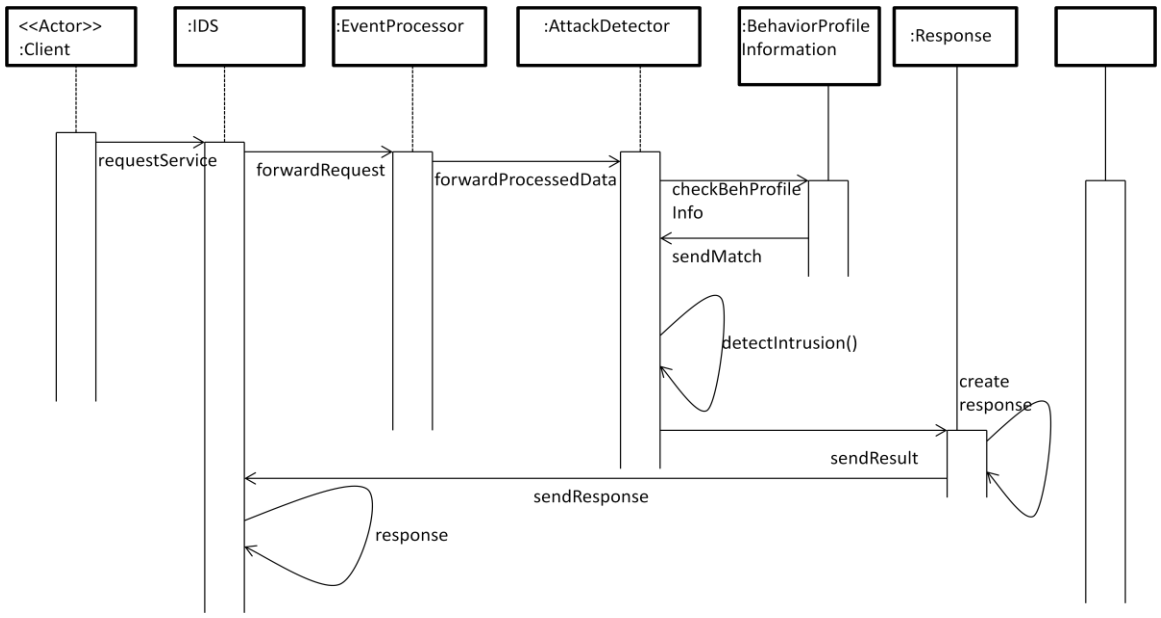


Figure 4.8 Sequence Diagram for detecting an intrusion with Behavior-based IDS.

Summary: The client requests a service from the host. The Behavior-Based IDS intercepts the message and compares whether the behavior of the request matches a normal behavior profile, if it does not, an attack is suspected and a response is raised.

Actors: Client and Server.

Precondition: We have a set of normal behavior profiles available.

Description:

a) Client makes a service request to the host.

b) The IDS sends the request event to an Event Processor.

c) The Event Processor processes the event as required by the Attack Detector and passes the processed event data to the Attack Detector.

d) The Attack Detector tries to detect whether this request is an attack or not by comparing the behavior profile of the request with the available behavior profiles in the Behavior Profile Information.

f) If a match is detected a Response is created.

Alternate Flows:

1) The Behavior Profile Information may not be able to detect an attack (a false negative).

2) The Behavior Profile Information can match and may indicate an attack when no attack is present (a false positive).

3) If no attack is detected, the request for service is forwarded to the host.

Postcondition: If an attack is detected while happening, suitable preventive measures can be adopted.

Example Resolved

We added an Intrusion Detection System to our network. Now all traffic is checked against a normal behavior profile to see whether the access request is an anomaly and hence a possible attack. We are now able to detect many new attacks which do not have a known signature and prevent them.

Implementation

Examples of techniques used for anomaly detection in practice are:

- Genetic Algorithm: In this approach, applications are modeled in terms of different system calls for different conditions such as normal behavior, error conditions and

attack conditions. A typical genetic algorithm involves two steps. The first step involves coding the vectors with a string of bits, which forms the input population of the algorithm. The second step is finding a fitness function to test each individual of the population against some evaluation criteria. In the learning process each event sequence of node behavior forms a gene. Fitness is calculated for a collection of genes. If genes with required fitness cannot be found in the current generation, new sets of genes are evolved through crossover and mutation. The process of evolution continues until genes with the required fitness are found. The detection process involves defining vectors for event data and methods of testing whether the vector indicates an intrusion or not [Kis07].

- Protocol Verification: The basis for this approach is the fact that most intruders use irregular or unusual protocol fields, which are not handled properly by application systems [Ver02].
- Statistical Models: These can be either multivariate models or models based on available statistics such as threshold measures, mean and standard deviations of the profile. Clustering analysis where clusters represent similar activities or user patterns is also sometimes used [Ver02].

Known Uses

- Cisco IPS 4200 Series utilizes detection techniques including stateful pattern recognition, protocol parsing, heuristic detection, and anomaly detection [Cis00].

- AirTight's wireless IPS automatically detects, classifies, blocks and locates wireless threats using behavior analysis. They use a genetic algorithm to establish normal behaviors [Air].

Some other uses of Anomaly-based IDSs are given in Table 4.1 [Gar09].

Table 4.1 Network-based IDS platforms with anomaly detection functionalities, according to the manufacturer's information [Gar09].

Name	Manufacturer	Hybrid	Response	Anomaly-related techniques
AirDefense Guard	AirDefense, Inc.	Y(es)	Y	Detection, correlation and multi-dimensional detection
Barbedwire IDS Softblade	BarbedWire Technologies	Y	Y	Protocol analysis, pattern matching
BreachGate WebDefend™	Breach security	Y	N	Behaviour-based analysis, statistical analysis, Using correlation functions.
Bro	Lawrence Berkeley National Laboratory	Y	Y	Application level semantics, event analysis, pattern matching, protocol analysis

In the table the "*Hybrid*" column indicates hybrid detection, and the "Response" column indicates that some kind of response mechanism is also available

Consequences

This pattern has the following advantages:

- *New Attacks:* Detection can be effective against new attacks which could cause abnormal behavior in the network traffic. For example, we can identify an attack with a behavior such as, when a usually passive web server tries to connect to a large number of addresses it could be the result of a worm attack.
- *Real-Time:* This kind of IDS works well with network traffic that exhibits a normal behavior where it will be easier to detect an abnormal behavior for the network.
- *Increased Vulnerability:* This kind of IDS is usually good in wireless networks that are more vulnerable due to their mobile nature.

This pattern has the following liabilities.

- This technology leads to generation of lots of false positives. Many anomalies detected are not attacks but could be just unusual behaviors of previous users.
- This technology cannot be implemented in networks that do not have a predictable traffic pattern.
- The technology adopted for one network is not easily portable to another system and can be different from system to system in a network, as normal behavior for one system is usually not the normal behavior for another system.

- If the attacker does an attack mimicking regular traffic or normal behavior, the attack may go undetected.

Related Patterns

- This pattern is used in conjunction with the Signature-Based IDS pattern.
- Firewalls are usually used along with the IDS in a network. Hence the patterns for firewalls [Sch06] complement this pattern.
- The response class could be implemented as a Strategy pattern [Gam94].

Conclusion

We have written a pattern for the abstract IDS and also patterns for the Signature-Based IDS and the Behavior-Based IDS. These patterns are defined by their class diagrams and the sequence diagrams. These patterns can be catalogued and used as reference for a designer trying to find an appropriate defense for her system.

5 SECURITY PATTERNS FOR THE NETWORK PROTOCOLS

IPSec is a protocol suite for securing Internet Protocol (IP) communications by authenticating and encrypting IP packets of a communication session. IPSec also includes sub-protocols for establishing mutual authentication between the participating peers at the beginning of the session and negotiation of cryptographic keys to be used during the session. We present here a pattern to describe its security properties. This pattern is part of a set of patterns that describe network security mechanisms, which is part of a catalog of security patterns [Fer13].

TLS is a connection-oriented protocol that provides a secure channel between a client and a server at the transport layer. The protocol supports message confidentiality, message integrity, and client/server authentication. The TLS protocol also provides a means for the negotiation of security parameters, such as the encryption algorithms, encryption keys, hashing functions, etc., that are used to transmit data securely. We present here a pattern to describe its security properties. This pattern also has become a part of the catalog of security patterns [Fer13]. We first present an abstract pattern from which we derive concrete patterns for TLS and IPSec protocols.

5.1 Abstract Pattern for Protocols at Network Layers

Intent

Provide a secure channel between a client and a server where application messages are being communicated over one of the network layers. The client and the server can mutually authenticate and a secure channel can be established between them.

Example

A bank has two of its branch offices in two different cities. These branches need to connect with each other in a secure manner. Most of the communication between the two offices is sensitive and needs to be protected.

Context

Users are using applications through the public internet, where multiple networks are connected with each other through gateways. This is an insecure network.

Problem

The messages communicated between the nodes at the network or transport layer are vulnerable to attack by intruders who may try to read or modify them. Both the hosts may be impostors.

The solution to this problem is affected by the following **forces:**

- *Confidentiality:* The message transferred in the layer between the nodes could be intercepted and read. We need to avoid this attack.

- *Integrity:* The message communicated in the layer between the nodes could be intercepted and modified. We need to avoid this form of attack also.
- *Authenticity:* Either node could be an impostor, which may result in security breaches. A Man in the Middle attack is also possible where an attacker poses as somebody he is not.
- *Flexibility:* The algorithms used by the security protocol should be flexible and configurable to be able to handle new attacks and adjust the degree of security.
- *Transparency.* The security measures of the protocol should be transparent to the users.

Solution

Establish a cryptographic secure channel between the nodes. Provide means for host node and server node to authenticate each other. The nodes can negotiate what cryptographic algorithms they will use for communication based on hardware and security needs.

Structure

Figure 5.1 shows the class diagram for the abstract pattern for network layer protocols. A **Server Node** requests some **Service** from the **Client Node**. The **Protocol Controller** conveys this request using an **Authenticator** to authenticate both the nodes and creating a **Secure Channel** between them. The Authenticator and the Secure Channel are known patterns (See Related Patterns).

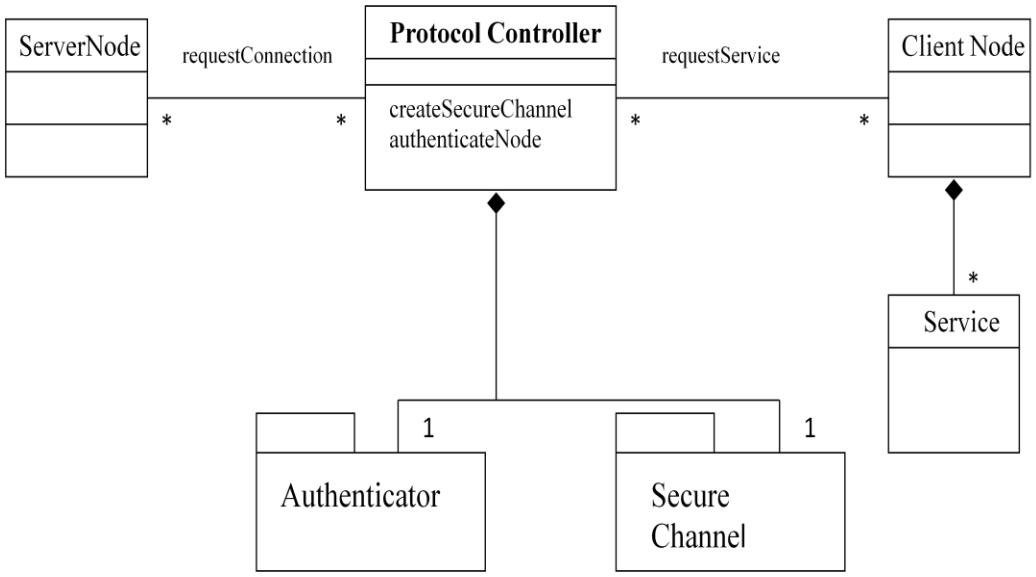


Figure 5.1 Class Diagram for Abstract Pattern for Network Layer Protocols.

Dynamics

We describe the dynamic aspects of the abstract Pattern using a sequence diagram for the following use case:

Request a service (Figure 5.2):

Summary: A node requests a service from client node on one of the network layers and the protocol that operates on that layer authenticates the request and creates a secure channel for communication of data.

Actors: Server Node, Client Node.

Precondition: The security parameters of the secure exchange have been predefined.

Description:

a) The Server Node makes a connection request to the Client Node on the network.

b) The protocol mutually authenticates the nodes to each other.

c) The protocol creates a secure channel between the nodes.

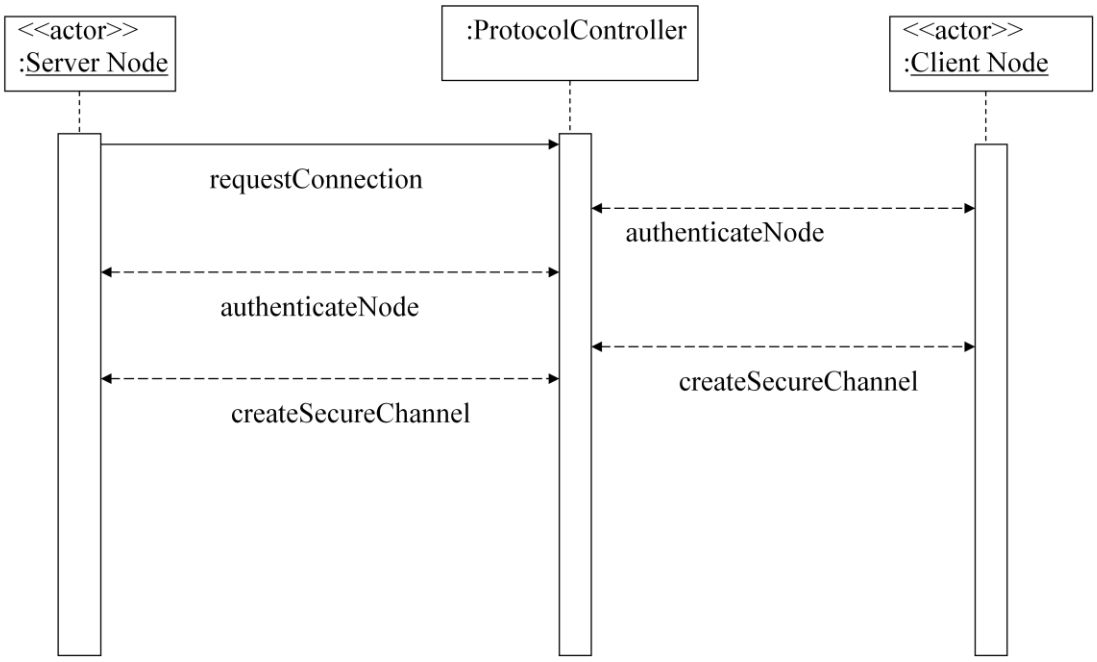


Figure 5.2 Sequence diagram for Use Case: Request a Service.

Alternate Flows:

1) The authentication can fail.

2) The creation of a secure channel can fail.

Postcondition: The client node accepts the request from service and is ready to start communication between the client and the server nodes.

Example Resolved

When two branches of the bank communicate with each other, each branch is authenticated to the other, and a secure channel is created between them. Now the bank branches know that the transactions are secure and the host servers of these banks can access the servers of the other one securely.

Variants or Concrete patterns

IPSec is the protocol used at the IP or network layer and TLS is the protocol implemented at the transport layer. These two protocols are developed into two concrete patterns in the next sections in this chapter.

Consequences

This pattern has the following advantages:

- *Confidentiality and integrity:* A secure channel is established between the nodes, which can provide data confidentiality and integrity for the messages sent. We could add a logging system for the client at its end point for future audits.
- *Authenticity:* Both the nodes that are communicating can be mutually authenticated. Man in the middle attacks can be prevented by mutual authentication.
- *Flexibility:* We can easily change encryption algorithms and authentication methods.
- *Transparency.* The users don't need to perform any operation to have a secure channel established.

This pattern has the following disadvantages:

- The protocol pattern adds some overhead to the communication.

.

Related Patterns

- The Authenticator describes how to mutually authenticate two nodes in a network [Hay00].
- The Secure Channel describes a cryptographic channel used to communicate secure data [Bra98].

5.2 Internet Protocol Security (IPSec)

Intent

Provide a secure channel between a client and a server where application messages are being communicated as IP packets over the internet layer. The client and the server are mutually authenticated and a secure channel is established between them.

Example

A bank has two of its branch offices in two different cities. These branches need to connect with each other in a secure manner. Most of the communication between the two offices is sensitive and needs to be protected.

Context

Users are using applications through public internet, where multiple networks are connected with each other through gateways. Packets are travelling between nodes

through various known and unknown networks. The network layer provides node to node communication services for these packets.

Problem

The messages communicated between nodes at the network layer are vulnerable to attack by intruders who may try to read or modify them. Both the hosts may be impostors.

The solution to this problem is affected by the following **forces:**

- *Confidentiality:* The message transferred in the layer between the nodes could be intercepted and read. We need to avoid this.
- *Integrity:* The message communicated in the layer between the nodes could be intercepted and modified. This needs to be prevented from happening.
- *Authenticity:* Either node could be an impostor, which may result in security breaches. A Man in the Middle attack is also possible where an attacker poses as somebody he is not.
- *Flexibility:* The algorithms used by the security protocol should be flexible and configurable to be able to handle new attacks.
- *Transparency.* The security measures of the protocol should be transparent to the users.

Solution

Establish a cryptographic secure channel between the nodes. Provide means for host node and server node to authenticate each other. The nodes can negotiate what

cryptographic algorithms they will use for communication based on hardware and security needs.

Structure

Figure 5.3 shows the class diagram for the basic architecture of the IPSec pattern. A **Server Node** requests some **Service** from the **Client Node**. The **IPSec Protocol** conveys this request using an **Authenticator** to authenticate both the nodes and creating a **Secure Channel** between them. The Authenticator and the Secure Channel are known patterns (See Related Patterns).

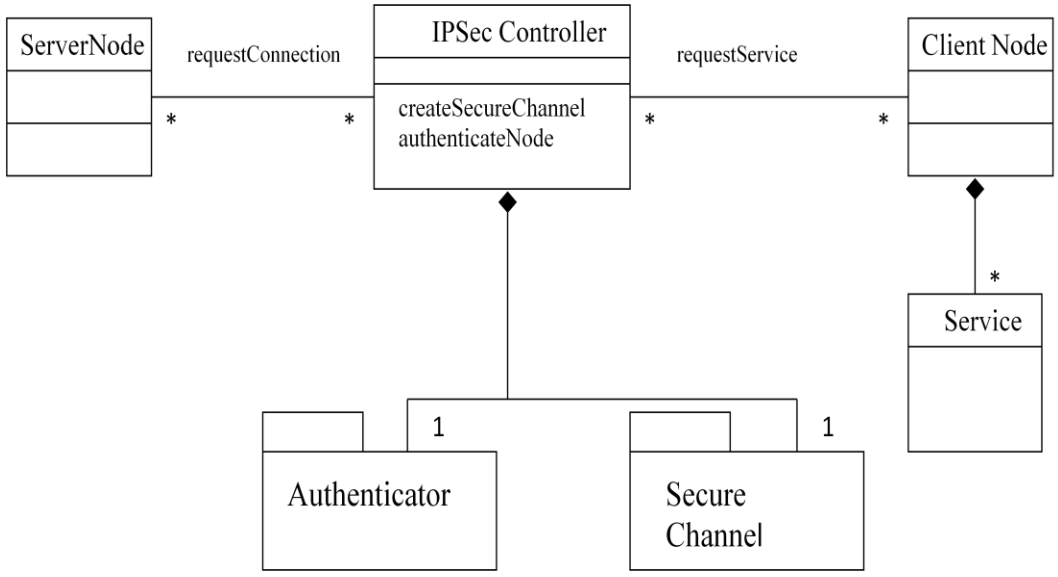


Figure 5.3 Class diagram for the IPSec Protocol pattern.

Dynamics

We describe the dynamic aspects of the IPSec Pattern using a sequence diagram for the following use case:

Request a service (Figure 5.4):

Summary: A node requests a service from client node on the network and the IPSec protocol authenticates the request and creates a secure channel for communication of packets.

Actors: Server Node, Client Node.

Precondition: The security parameters of the secure exchange have been predefined.

Description:

a) The Server Node makes a connection request to the Client Node on the network.

b) The IPSec protocol mutually authenticates the nodes to each other.

c) The IPSec protocol creates a secure channel between the nodes.

Alternate Flows:

1. The authentication can fail.
2. The creation of a secure channel can fail.

Postcondition: The client node accepts the request from service and is ready to start packet communication.

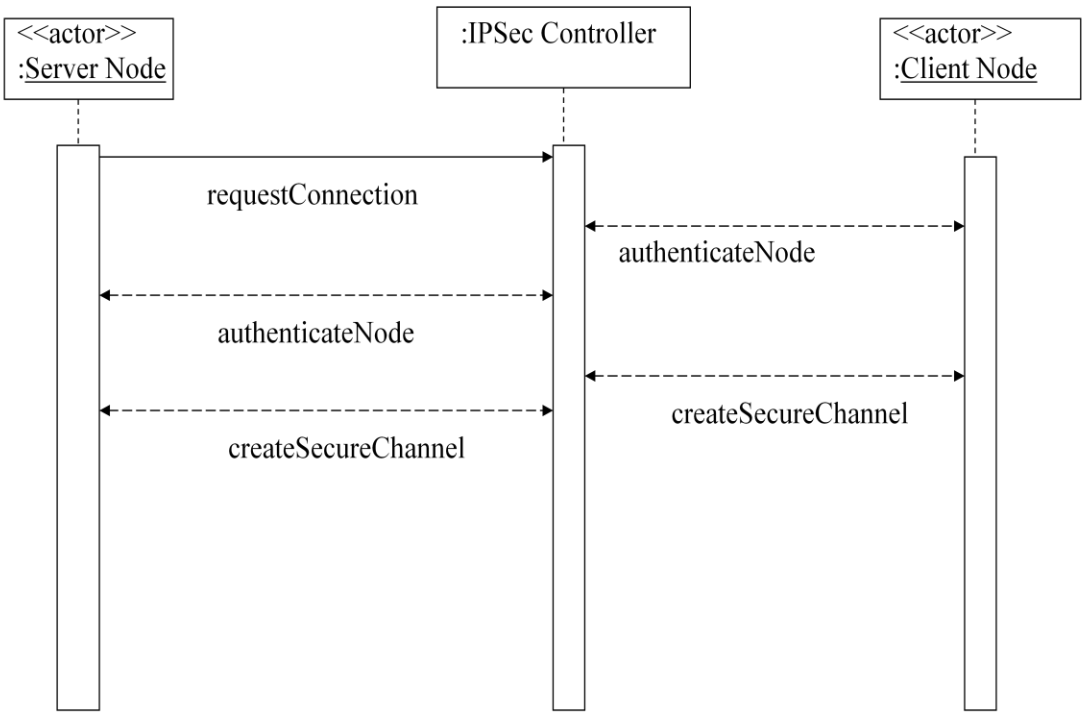


Figure 5.4 Sequence diagram for Use Case: Request a Service.

Implementation

This section includes the implementation details.

IPSec protocol suite consists mainly of an **Authentication Header** (AH) as shown in Figure 5.5. and **Encapsulating Security Payload** (ESP) as shown in Figure 5.6. The details are given below:

Authentication Header (AH) is a member of the IPsec protocol suite. AH guarantees connectionless integrity and data origin authentication of IP packets. Further, it can optionally protect against replay attacks by using the sliding window technique and

discarding old packets. In IPv6, the AH protects the most of the IPv6 base header, AH operates directly on top of IP, using IP protocol number 51.

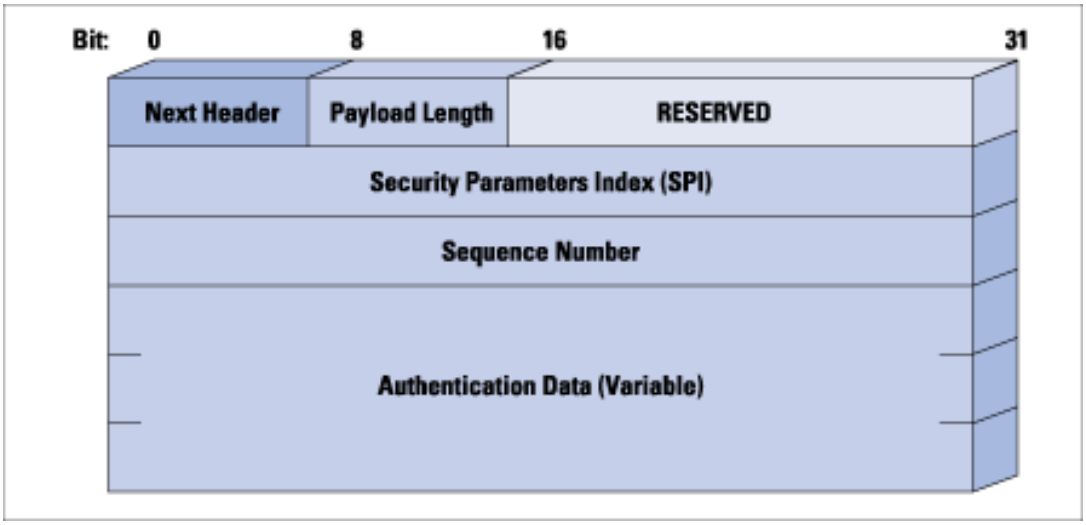


Figure 5.5 Authentication Header(AH) packet diagram[from Cis01]].

The terms in the diagram are explained below:

Next Header (8 bits)

Type of the next header, indicating what upper-layer protocol was protected. The value is taken from the list of IP protocol numbers.

Payload Length (8 bits)

The length of this *Authentication Header* in 4-octet units.

Reserved (16 bits)

Reserved for future use (all zeroes until then).

Security Parameters Index (32 bits)

Arbitrary value which is used (together with the destination IP address) to identify the security association of the receiving party.

Sequence Number (32 bits)

A monotonic strictly increasing sequence number (incremented by 1 for every packet sent) to prevent replay attacks. When replay detection is enabled, sequence numbers are never reused because a new security association must be renegotiated before an attempt to increment the sequence number beyond its maximum value.

Integrity Check Value (multiple of 32 bits)

Variable length check value. It may contain padding to align the field to an 8-octet boundary [WikIP].

Encapsulating Security Payload (ESP) is a member of the IPsec protocol suite. In IPsec it provides origin authenticity, integrity, and confidentiality protection of packets. ESP also supports encryption-only and authentication-only configurations, but using encryption without authentication is strongly discouraged because it is insecure. Unlike Authentication Header (AH), ESP in transport mode does not provide integrity and authentication for the entire IP packet. However, in Tunnel Mode, where the entire original IP packet is encapsulated with a new packet header added, ESP protection is

afforded to the whole inner IP packet (including the inner header) while the outer header remains unprotected. ESP operates directly on top of IP, using IP protocol number 50.

The following ESP packet diagram in Figure 5.6 shows how an ESP packet is constructed and interpreted.

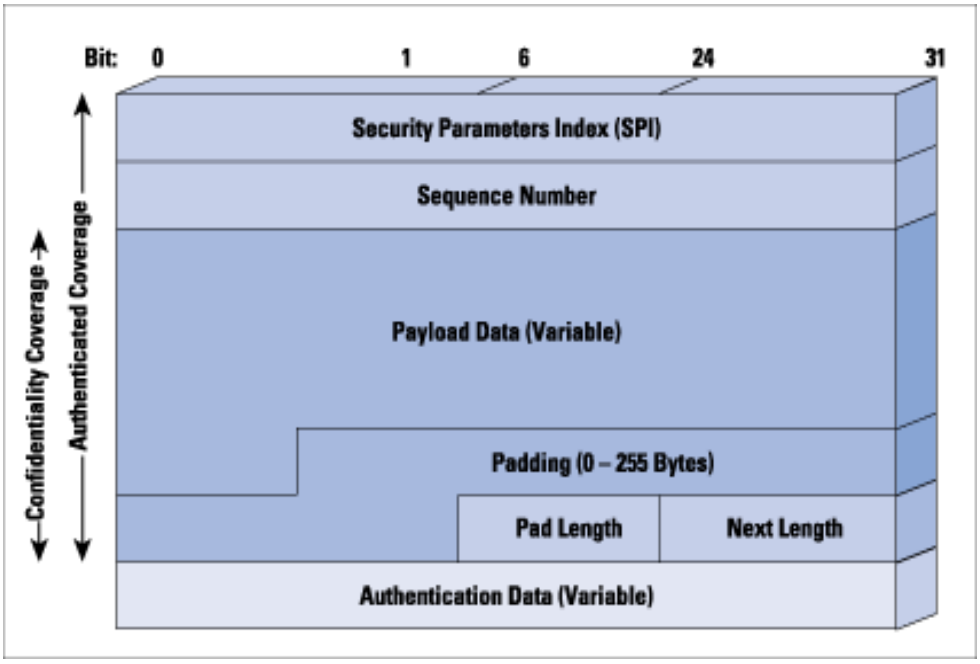


Figure 5.6 Encapsulating Security Payload (ESP) packet diagram [from Cis01].

The terms in the diagram are explained below:

Security Parameters Index (32 bits)

Arbitrary value used (together with the destination IP address) to identify the security association of the receiving party.

Sequence Number (32 bits)

A monotonically increasing sequence number (incremented by 1 for every packet sent) to protect against replay attacks. There is a separate counter kept for every security association.

Payload data (variable)

The protected contents of the original IP packet, including any data used to protect the contents (e.g. an Initialisation Vector for the cryptographic algorithm). The type of content that was protected is indicated by the *Next Header* field.

Padding (0-255 octets)

Padding for encryption, to extend the payload data to a size that fits the encryption's cipher block size, and to align the next field.

Pad Length (8 bits)

Size of the padding (in octets).

Next Header (8 bits)

Type of the next header. The value is taken from the list of IP protocol numbers.

Integrity Check Value (multiple of 32 bits)

Variable length check value. It may contain padding to align the field to an 8-octet boundary [WikIP].

Variants

IPSec can be implemented in a host-to-host transport mode, as well as in a network tunnel mode.

Transport mode:

In transport mode, only the payload of the IP packet is usually encrypted and/or authenticated. The routing is intact, since the IP header is neither modified nor encrypted; however, when the authentication header is used, the IP addresses cannot be translated, as this will invalidate the hash value. The transport and application layers are always secured by hash, so they cannot be modified in any way (for example by translating the port numbers).

Tunnel mode:

In tunnel mode, the entire IP packet is encrypted and/or authenticated. It is then encapsulated into a new IP packet with a new IP header. Tunnel mode is used to create virtual private networks for network-to-network communications (e.g. between routers to link sites), host-to-network communications (e.g. remote user access), and host-to-host communications (e.g. private chat).

Known Uses

- IPSec is supported by the Windows Server 2003, Windows XP, and Windows 2000 operating systems and is integrated with the Active Directory service. IPSec policies can be assigned through Group Policy configuration of Active Directory domains and organizational units. This allows the IPSec policy to be assigned at the domain, site, or organizational unit level, simplifying IPSec deployment [MS].
- IBM implements VPN concepts using IP security (IPSec) and Layer 2 Tunneling Protocol (L2TP) on the AS/400 operating system [IBM99].
- HP-UX IPSec (J4256AA) provides an infrastructure for secure communications over IP-based networks between systems and devices that implement the IPsec protocol suite [HP].

Example Resolved

When two branches of the bank communicate with each other, each branch is authenticated to the other, and a secure channel is created between them. Now the bank branches know that the transactions are secure and the host servers of these banks can access the servers of the other one securely.

Consequences

This pattern has the following advantages:

- *Confidentiality and integrity:* A secure channel is established between the nodes, which can provide data confidentiality and integrity for the messages sent. We could add a logging system for the client at its end point for future audits.

- *Authenticity:* Both the nodes that are communicating can be mutually authenticated. Man in the middle attacks can be prevented by mutual authentication.
- *Flexibility:* We can easily change encryption algorithms and authentication methods.
- *Transparency.* The users don't need to perform any operation to have a secure channel established.

This pattern has the following disadvantages:

- The protocol adds some overhead to the communication.
- IPSec also has the disadvantage of requiring operating system support, since most operating systems don't allow direct manipulation of IP headers.

Related Patterns

- The Authenticator describes how to mutually authenticate two nodes in a network [Bro99].
- The Secure Channel describes a cryptographic channel used to communicate secure data [Bra98]
- *IPsec VPNs* provide a secure, network-layer connection to the corporate network. As data traverses the Internet from the mobile device to the VPN gateway, it is encapsulated and encrypted. After the traffic passes through the VPN gateway and onto the LAN, it is no different from traffic coming directly from end users on the LAN [Kum10].
- *TLS Patterns* [Kum12b] provide a secure channel between a client and a server where application messages are being communicated over the Transport layer of the Internet.

5.3 Transport Layer Security (TLS)

Intent

We need to provide a secure channel between a client and a server where application messages are being communicated over the Transport layer of the Internet. The client and the server need to be mutually authenticated.

Example

A bank customer may want to check his account balance online. The bank uses the Transport layer to transfer its confidential data. We need to protect this communication as this confidential data is vulnerable to attacks. The customer also has to make sure that the transactions are made with the bank and not with an impostor, while the bank may need to verify that this is a legitimate customer.

Context

Users using applications that exchange sensitive information using web browsers for e-commerce or similar activities. The **transport layer** provides end-to-end communication services for applications within a layered architecture of network components and protocols. The transport layer provides convenient services such as connection-oriented data stream support, flow control, and multiplexing.

Problem

The messages communicated between applications and servers at the transport layer are vulnerable to attack by intruders who may try to read or modify them. The server and the client may be impostors.

The solution to this problem is affected by the following **forces:**

- *Confidentiality and integrity:* The data transferred in the transport layer between the client and the server could be intercepted and read or modified illegally. This should be avoided.
- *Authenticity:* Either the server or the client could be an impostor, which may allow security breaches. A Man in the Middle attack is also possible where an attacker poses as the client to the server and as the server to the client.
- *Flexibility:* The algorithms used in the security protocol should be flexible and configurable to be able to handle new attacks.
- *Transparency.* The security measures of the protocol should be transparent to the users.
- *Configurability:* The protocol should allow the users to select different algorithms to provide different degrees of security.

Solution

Establish a cryptographic secure channel between the client and the server at the transport layer. Provide means for client and server to authenticate each other. The client and server can negotiate what cryptographic algorithms they will use.

Structure

Figure 5.7 shows a class diagram for the basic architecture of the TLS pattern. A **Client** requests some **Service** from the **Server**. The **TLS Protocol** conveys this request using an **Authenticator** to authenticate both the Server and the Client and creating a

Secure Channel between the client and the server. The Authenticator and the Secure Channel are known patterns (See Related Patterns).

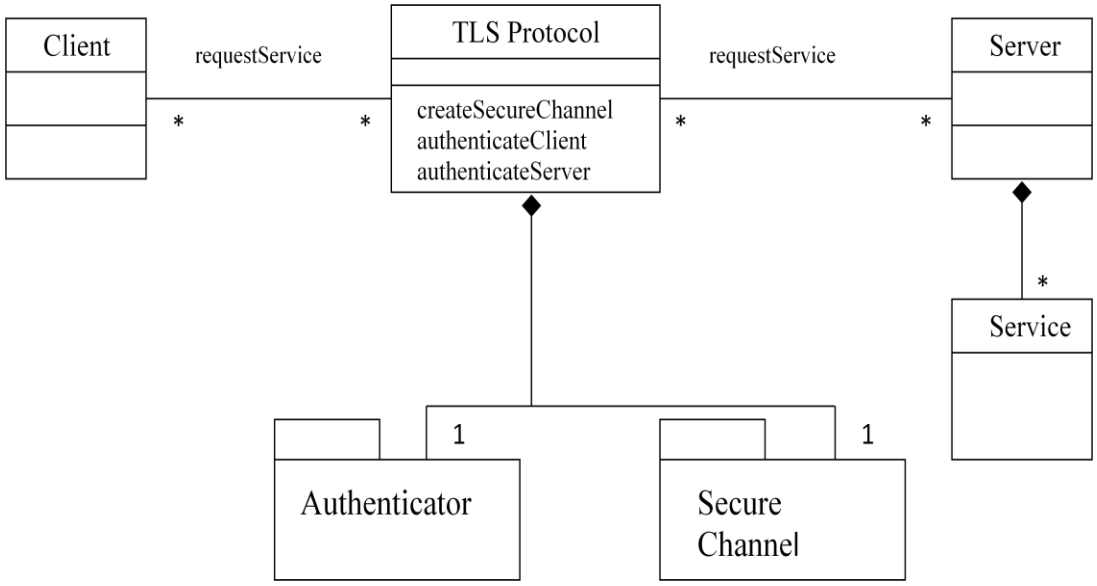


Figure 5.7 Class diagram for the TLS Protocol pattern.

Dynamics

We describe the dynamic aspects of the TLS Pattern using a sequence diagram for the following use case:

Request a service (Figure 5.8):

Summary: A client requests a service and the TLS protocol authenticates the request and creates a secure channel.

Actors: Client, Server.

Precondition: The security parameters of the secure exchange have been predefined.

Description:

a) The client makes a service request to the server.

b) The TLS protocol authenticates the server to the client and the client to the server.

c) The TLS protocol creates a secure channel between the server and the client.

Alternate Flows:

1. The authentication can fail.
2. The creation of a secure channel can fail.

Postcondition: The server accepts the request and grants the service.

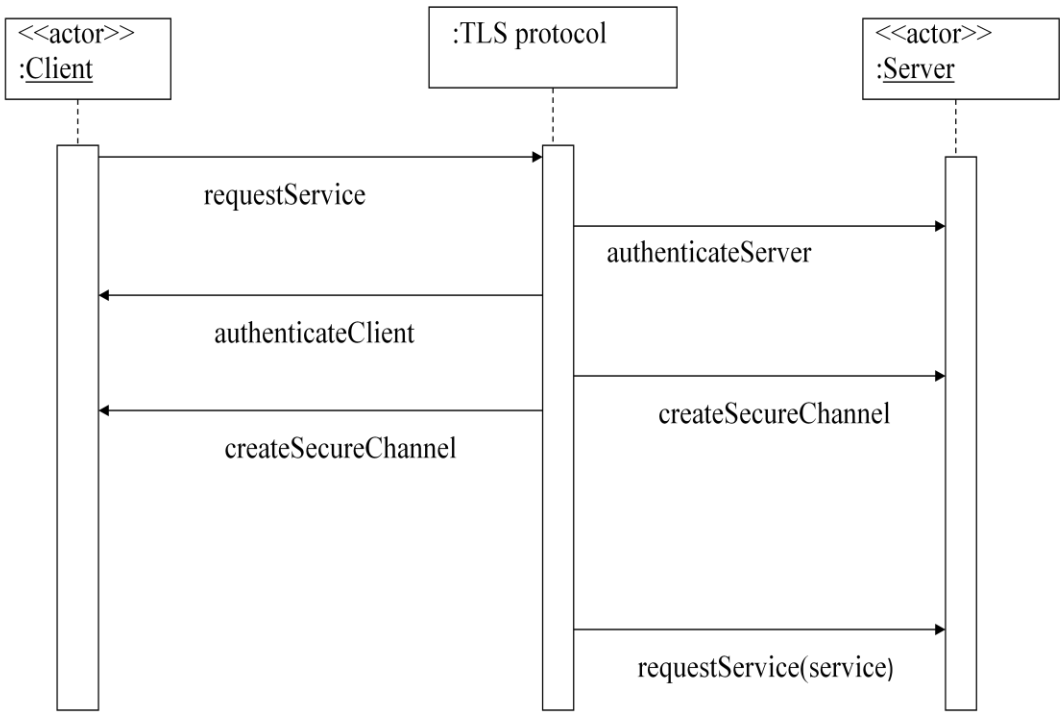


Figure 5.8 Sequence diagram for Use Case: Request a Service.

Implementation

One of the protocols that is dominant today for providing security at the transport layer is the Secure Sockets Layer (SSL) Protocol. The SSL protocol is a transport layer security protocol that was developed and proposed by Netscape Communications in the 1990s. The Transport Layer Security (TLS) Protocol is an IETF version of the SSL protocol, which has become a standard [Yas04]. A good amount of implementation advice can be found in [Sel12].

The TLS protocol is mainly partitioned into two protocol layers, the *TLS Record Protocol* and the *TLS Handshake Protocol*, executing above the TCP (Transport Layer) protocol as shown in Figure 5.9 [Elg06, Sta12]. There are other minor protocols at the

handshake protocol layer such as the *Cipher Change Protocol*, *Alert Protocol*, and *Application Protocol*.

<table>
<tr><td>TLS Handshake Protocol</td><td>TLS Cipher change Protocol</td><td>Alert Protocol</td><td>Application Protocol</td></tr>
<tr><td colspan="4">TLS Record Protocol</td></tr>
<tr><td colspan="4">TCP</td></tr>
<tr><td colspan="4">IP</td></tr>
</table>

Figure 5.9 TLS Protocol layers.

Record Protocol

The TLS Record Protocol provides encryption and message authentication for each message. A connection is created using symmetric cryptography data encryption. The keys for this symmetric encryption are generated uniquely for each connection and are based on a secret negotiated by another protocol (such as the TLS Handshake Protocol). Messages include a message integrity check using a keyed MAC, computed using hash functions [Sta03].

Handshake Protocol

A TLS *handshake* supplies the authentication and key exchange operations for the TLS protocol. The security state agreed upon in the handshake is used by the TLS Record Protocol to provide session security. This protocol allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives any data. The TLS Handshake Protocol provides connection security where the peers' identities can be authenticated using asymmetric cryptography. This authentication can be made optional, but is generally required for at least one of the peers.

A TLS session is an association between a client and a server, created by the handshake protocol. Sessions define a set of cryptographic security parameters, which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.

A session state is defined by the following parameters:

- *Session identifier*. It is generated by the server to identify a session with a chosen client.
- *Peer certificate*. X.509 certificate of the peer.
- *Compression method*. A method used to compress data prior to encryption.
- *Algorithm specification or CipherSpec*. Specifies the encryption algorithm that encrypts the data and the hash algorithm used during the session.

- *Master secret*: 48-byte data being a secret shared between the client and server, "it *is resumable*": a flag indicating whether the session can be used to initiate new connections

The Handshake protocol consists of the following four phases:

a) In the first phase, an initial connection is established which starts the negotiation. The client and server exchange hello messages that are used to establish security parameters (as defined above) used in the TLS session and settings used during the handshake, such as the key exchange algorithm.

b) During the second phase (authentication), the server sends a Certificate message to the client that may include a server certificate when an RSA key exchange is used, or Diffie-Hellman parameters when a Diffie-Hellman key exchange is used. The server may also request a certificate from the client, using the certificateRequest message.

c) During the third phase, the client, if asked, may send its certificate to the server in a Certificate message along with a certificateVerify message so that the server can verify certificate ownership if the server requested a client certificate during the second phase. This phase includes the establishment of the security parameters such as the encryption key. The client must send either a pre-master secret encrypted using the server's public key, or public Diffie-Hellman parameters in the clientKeyExchange message so that the client and server can compute a shared master secret.

d) In the fourth phase, the client and server finish the handshake, which implies that the client and server are mutually authenticated and have completed the required key exchange operations.

Structure and Dynamics of the Handshake Protocol

We describe the structure of the Handshake protocol using the class diagram in Figure 5.10. The **Client** requests for a service from the **Server** at the Transport layer. The TLS **Handshake Protocol** uses **Certificate(s)** to mutually authenticate the Client and the Server and does the clientKeyExchange once the authentication is done.

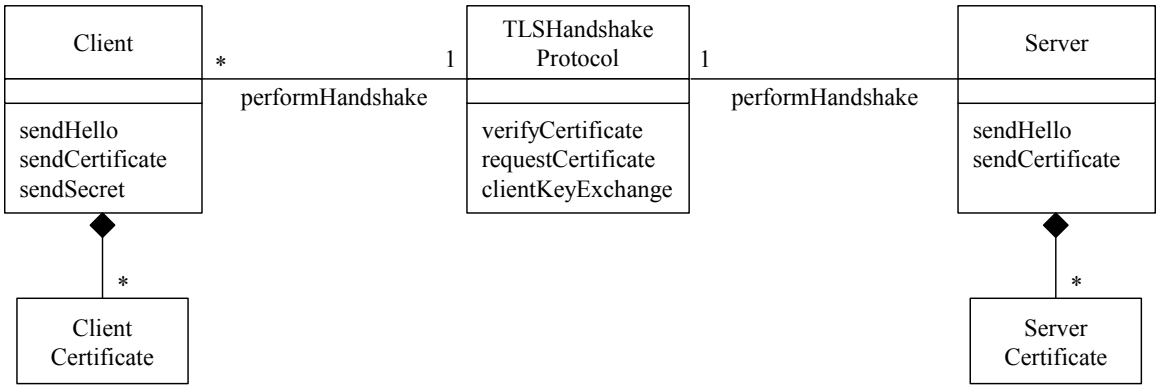


Figure 5.10 Class diagram for the TLS handshake protocol.

We describe the dynamic aspects of the TLS Handshake using a sequence diagram (Figure 5.11).

Summary: A TLS *handshake* supplies the authentication and key exchange operations for the TLS protocol.

Actors: Client, Server.

Precondition: The client has made a request for a service from the host server and an initial connection has already been established. The client and server need to have a digital certificate, issued by some Certificate Authority,

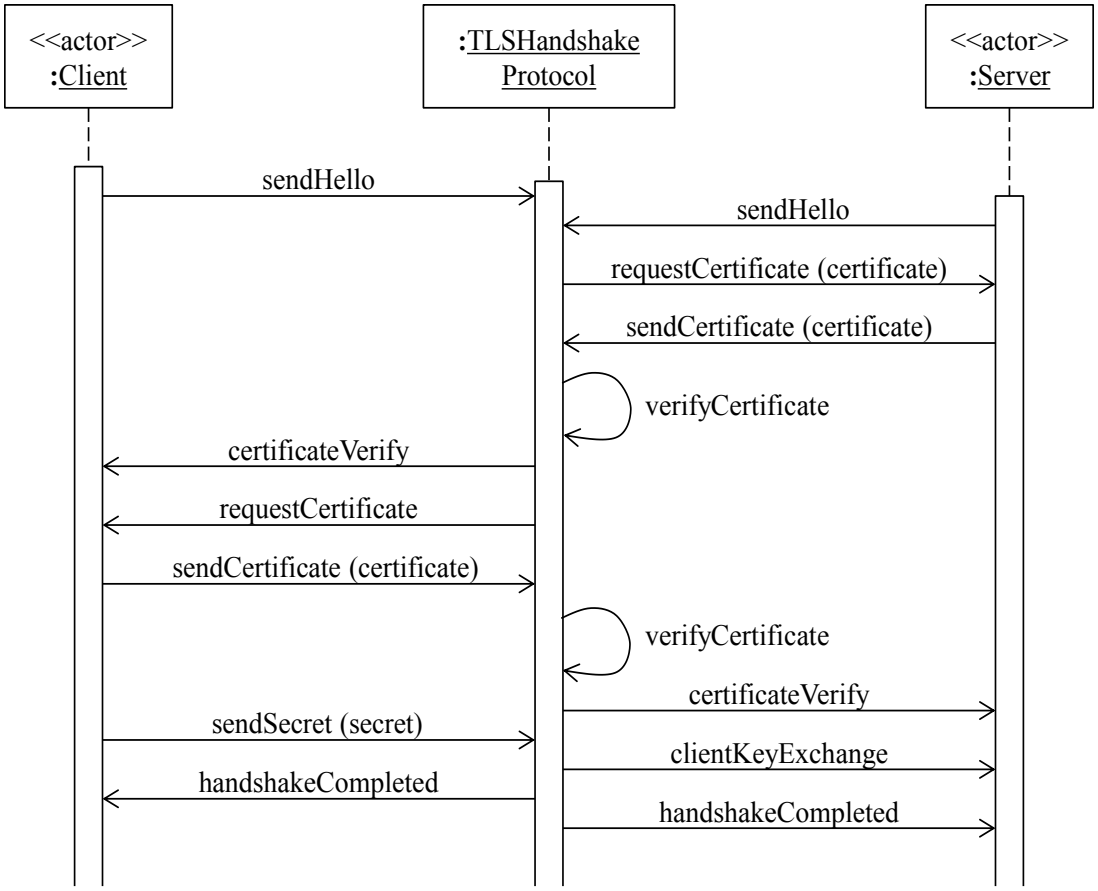


Figure 5.11 Sequence diagram for the TLS handshake use case

Description:

a) The client and server exchange initial Hello messages.

b) The protocol requests the certificate from the server and the server sends the certificate.

c) The server certificate is verified

d) The server requests the certificate from the client (optional).

e) If asked, client sends the certificate to the server.

f) The client certificate is verified.

g) The client sends the predefined secret encrypted using the server's public key which is the client key exchange.

h) The client and the server complete mutual handshake and the initial encryption parameters.

Alternate Flows:

1) Authentication of the server or client can fail. A certificate can be expired or outdated.

2) The client could lose the encryption key while exchanging with the server.

Postcondition: Client and Server can start exchanging data at the transport layer.

The other minor protocol layers from Figure 5.9 are discussed below:

Cipher Change Protocol

This protocol signals transitions in cipherSpec which is a session parameter explained above.

Alert Protocol

This protocol raises alerts for the communication. This record should normally not be sent during normal handshaking or application exchanges. However, this message can be sent at any time during the handshake and up to the closure of a TLS session. If this record is used to signal a fatal error, the session will be closed immediately after sending this record. If the alert level is flagged as a warning, the remote partner can decide or not to close the session.

Application Protocol

Now the handshake is completed and the Application protocol is enabled. This marks the start of data exchange between the server and the client.

Variants

- *WTLS:* A modified version of TLS, called WTLS (Wireless TLS protocol) has been used in mobile systems. WTLS is based on TLS and is similar in some aspects [Bad04]. WTLS has been superseded in the WAP (Wireless Aplication Protocol) 2.0 standard by the End-to-end Transport Layer Security Specification.

- *MultipleTLS (MTLS):* This is an application-level protocol running over the TLS Record protocol. The MTLS provides application multiplexing over a single TLS

session. Therefore, instead of associating a TLS session with each application, this protocol allows several applications to protect their communication over a single TLS session [Bad09].

Some different versions of TLS are given below:

TLS1.0: TLS 1.0 was first defined in January 1999 as an upgrade to SSL Version 3.0 and is an IETF version of SSL. The differences between this protocol and SSL 3.0 are not large, but they are significant enough that TLS 1.0 and SSL 3.0 do not interoperate. TLS 1.0 does include a means by which a TLS implementation can downgrade the connection to SSL 3.0, but this weakens security.

TLS1.1: TLS 1.1, defined in April 2006, is an update of TLS version 1.0. Significant differences include:

- Added protection against Cipher Block Chaining (CBC) attacks. (In CBC mode, each block of plaintext is XORed with the previous cipher text block before being encrypted.)
    - The implicit Initialization Vector (IV) was replaced with an explicit IV.
    - Change in handling of padding errors.
- Support for registration of parameters.

TLS1.2: This is a revision of the TLS 1.1 protocol from August 2008, which contains improved flexibility, particularly for negotiation of cryptographic algorithms. The major changes are:

- The MD5/SHA-1 combination in the pseudorandom function (PRF) has been replaced with cipher-suite-specified PRFs. All cipher suites in this document use P_SHA256.
- The MD5/SHA-1 combination in the digitally-signed element has been replaced with a single hash. Signed elements now include a field that explicitly specifies the hash algorithm used.
- Substantial cleanup to the client's and server's ability to specify which hash and signature algorithms they will accept.
- Addition of support for authenticated encryption with additional data modes.
- Tightening up of a number of requirements.
- Verification of data length now depends on the cipher suite (default is still 12) [WikTLS].

Known Uses

- Mozilla Firefox versions 2 and above, support TLS 1.0 [Moz].
- Internet Explorer(IE) 8 in Windows 7 and Windows Server 2008 support TLS 1.2 [MS].
- Presto 2.2, used in Opera 10, supports TLS [Ope].

Example Resolved

When a request is made to the bank server by the online client at the transport layer the bank server is authenticated to the customer, the customer is authenticated to the server and a secure channel is created between them. Now the client knows that the online bank transactions are secure.

Consequences

This pattern has the following advantages:

- *Authentication:* Both client and server can be mutually authenticated. Man in the middle attacks can be prevented by mutual authentication.
- *Security:* A secure channel is established between the server and the client, which can provide data confidentiality and integrity for the messages sent. We could add a logging system for the client at its end point for future audits.
- *Flexibility:* We can easily change encryption algorithms and authentication methods.
- *Transparency.* The users don't need to perform any operation to have a secure channel established.

This pattern has the following disadvantages:

- The protocol adds some overhead to the communication.
- SSL/TLS is a two-party protocol, thus it is not designed to handle multiple parties. However, as described earlier, the MTLS variant can handle multiple parties.

Related Patterns

- The Authenticator describes how to mutually authenticate a Client and a Server [Bro99].
- The Secure Channel describes a cryptographic channel used to communicate secure data [Bra98]
- The **SSL VPNs** provide a secure, transport-layer connection to the corporate network and is most commonly deployed for new enterprise remote access deployments [Kum10].

- The **IPSec Pattern** describes the pattern for the IPSec protocol.

Conclusion

We have written a pattern for the abstract Network Layer Protocol and also patterns for the IPSec protocol and the TLS protocol. These patterns are defined by their class diagrams and the sequence diagrams.

6 SECURITY PATTERNS FOR VIRTUAL PRIVATE NETWORKS

The Internet Protocol Suite, also referred to as TCP/IP, defines a reference model for networks that includes four layers [Sta03]: Application, Transport, Internet, and Link. One can apply security to any of these layers. Two secure protocols are commonly used in networks:

- The IPSec protocol, which provides cryptographic functions at the Internet (IP) layer [For04, Sta03].
- The Transport Layer Security (TLS) protocol, which provides similar functions at the Transport (TCP) layer [For04, Sta03]. This protocol is based on the Secure Sockets Layer (SSL) protocol.

One can also apply security defenses at the network boundaries, where networks enter the computational nodes. Two security mechanisms are normally used in network boundaries:

- Firewalls, which filter input and output traffic according to predefined rules. There are already patterns available for firewalls at the IP (Packet filter) and TCP (Proxy Firewall) layers [Sch06], as well as the (User) Application layer [Del04].
- Intrusion Detection Systems (IDS) try to detect attacks in real time. We already have seen patterns for Signature-Based and Behavior-Based IDS in Chapter 4.

A third type of mechanism is frequently used, the Virtual Private Network (VPN). Virtual Private Networks make use of public network resources to access the internal nodes of an enterprise. The transmission is protected through a cryptographic tunnel that provides message confidentiality and integrity. Since this network exists only in a virtual sense, it has been termed a virtual private network.

Figure 6.1 shows a pattern diagram that puts in perspective the different types of VPNs with respect to the network security protocols they use. The Abstract VPN pattern defines the basic functions and threats of a VPN, independently of the protocol where it operates. An abstract pattern defines only fundamental, implementation-independent functions and threats [Fer08]. Concrete patterns add functionalities and threats and take into account the characteristics of their specific concrete environment. In this case, the abstract functions are realized by concrete VPNs which operate according to the rules of specific protocols, i.e. IPSec VPN and TLS VPN. We present here patterns for the Abstract, IPSec VPN, and TLS VPN. Figure 6.1 shows also patterns for the protocols, TLS and IPSec, which in turn use patterns for Authentication [Sch06] and Secure Channel [Bra00] (not shown in the figure).

Section 6.1 describes an Abstract VPN pattern, defining the common features and threats of all VPNs. Sections 6.2 and 6.3 describe concrete VPNs for the IPSec and SSL respectively. For these concrete patterns, we show only their differences with respect to the abstract pattern, the assumption being that all their other aspects are inherited from the abstract pattern. Our patterns are intended for system designers, who can use these differences to select one type of pattern or the other. A product designer could use the

patterns as a guideline for their functionality. Finally, users and administrators of systems can use the patterns to understand their functions.

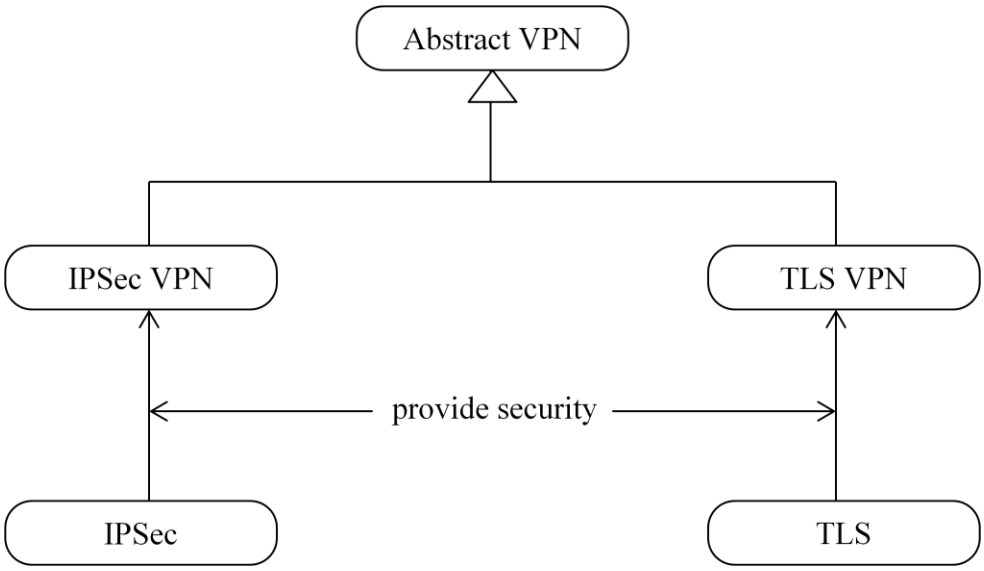


Figure 6.1 Pattern diagram for VPN patterns.

6.1 Abstract VPN

Intent

This pattern sets up a secure channel between two endpoints using cryptographic tunneling, with authentication at each endpoint. An endpoint is an interface exposed by a communicating unit (user site or network).

Example

Our company has employees all over the world and because of cost we decided to use the Internet to communicate. However, we are having problems because their orders are hacked and the attackers get customers' credit card numbers and other details. The employees want to be sure they are talking to other employees and should be able to send secure messages between them to discuss prices, discounts, and similar confidential information.

Context

Users scattered in many predefined locations, who need to communicate securely with each other, using the Internet or another insecure network. In such a network attacker may intercept messages and try to read, modify, or replay them.

Problem

In today's world, companies have offices all over the world and a lot of people work remotely. They need a secure connection to other specific nodes so that confidential work can be performed securely. Their communication can be intercepted by attackers who may get access to private information and may even modify the messages. How do we establish a secure channel for the end users of a network so they can exchange messages through some fixed points using an insecure network?

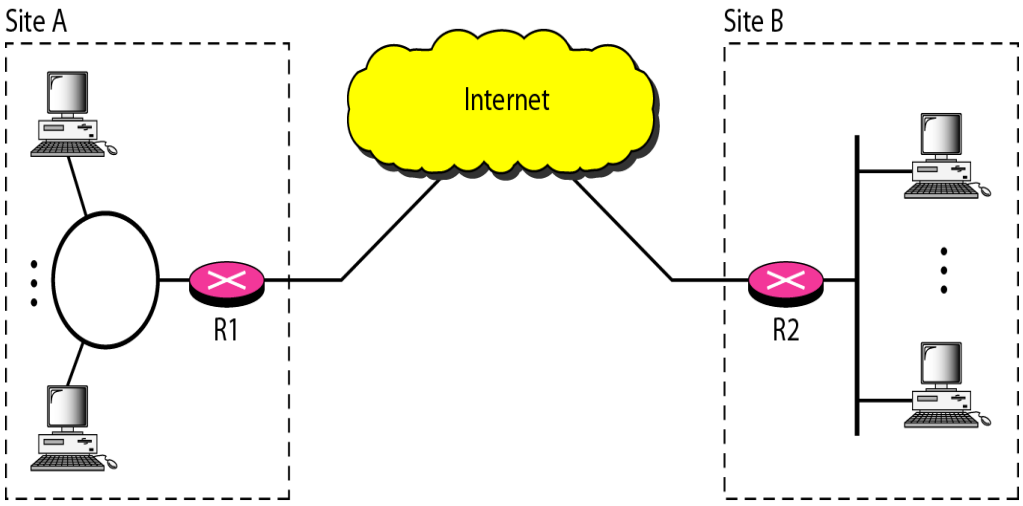
The solution to this problem is affected by the following **forces:**

- We need to use the Internet or other insecure networks to reduce the cost; in turn subjecting our network to numerous threats.

- Only registered users should access the institution's end points.
- We need to make sure that the users with which we are communicating are the right ones; otherwise confidentiality may be compromised.
- The number of users remotely connected may be growing; the system should be scalable.
- Because different users or institutions require different levels of security, the system should be flexible enough to accommodate different ways of providing security and different degrees of security.
- In some cases we also need to support authorization to access specific resources in the endpoints.
- The system should be easy to use and set up. Else, the users and administrators will be annoyed and will not want to use it.
- The system should not impose a heavy performance penalty. Otherwise, it will not be used all the time.
- The pattern should be adaptable to the needs and constraints of different protocol layers.

Solution

Protect communications by establishing a cryptographic tunnel between endpoints at one of the layers of the communication protocol. Add authentication functions at each endpoint. Figure 6.2 shows the case where Site A is talking to Site B over the Internet using routers R1 and R2, respectively. The secure connection is established through one of the Internet layers.



R1 – Router at Site A.

R2 – Router at Site B..

Figure 6.2 Two sites communicating through the Internet [from For04].

Structure

Figure 6.3 shows the class diagram for the VPN pattern where a **Secure Channel** can be established between a Client and a network **EndPoint**. EndPoints communicate with other EndPoints. A user is authenticated by an **Authenticator** pattern [Sch06]. Authenticator and Secure Channel are patterns, composed typically of several classes, and are indicated with the UML symbol for package.

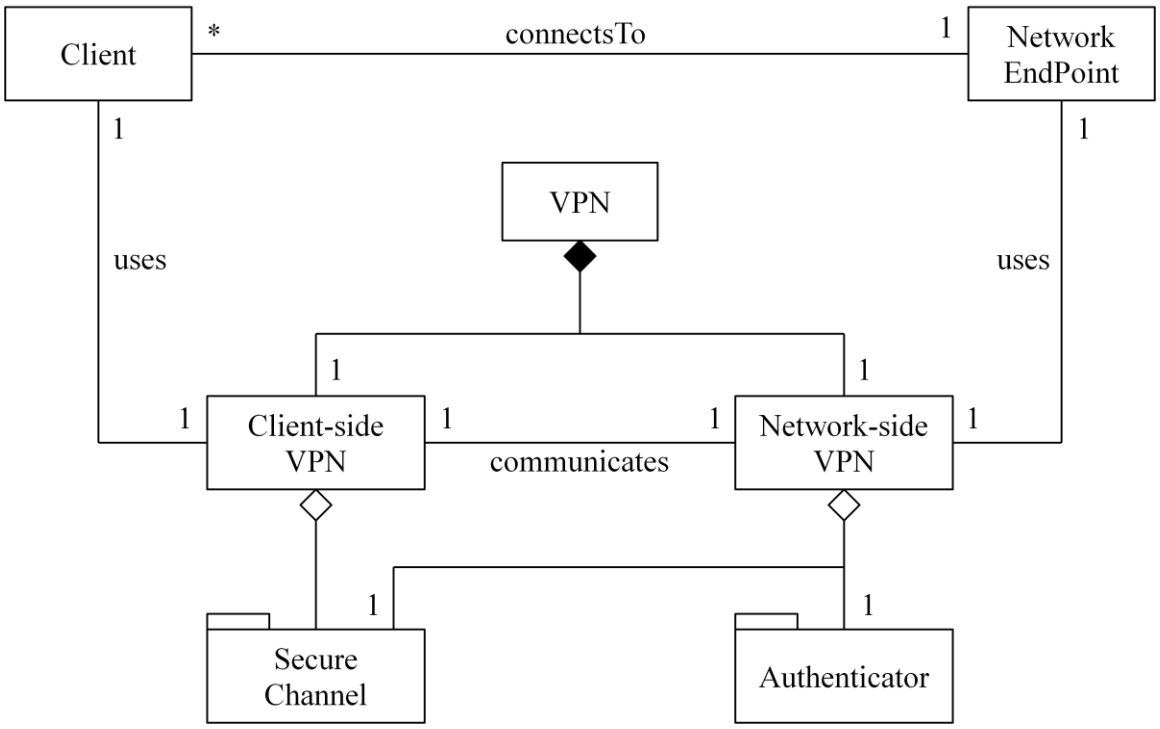


Figure 6.3 Class diagram for the abstract VPN security pattern.

Dynamics

The sequence diagram of Figure 6.4 shows a use case where an end user tries to access an endpoint in a network, endPoint2, from another endpoint, endPoint1. The Authenticator at endPoint1 authenticates the user. The Authenticator creates a Token as proof of authentication, which can be used to establish a Secure Channel. This channel allows secure access to endPoint2.

We describe the dynamic aspects of the abstract VPN pattern using a sequence diagram for the following use case:

User accessing another end point in a network (Figure 6.4):

Summary: An end user tries to access an endpoint in a network, endPoint2, from another endpoint, endPoint1.

Actors: User, endPoint1, endPoint2.

Precondition: None

Description:

a) The Authenticator at endPoint1 authenticates the user.

b) The Authenticator creates a Token as proof of authentication, which can be used to establish a Secure Channel.

c) This channel allows secure access to endPoint2.

Alternate Flows:

1. The authentication can fail.
2. The creation of a secure channel can fail.

Postcondition: The end user accesses endPoint2 from endPoint1.

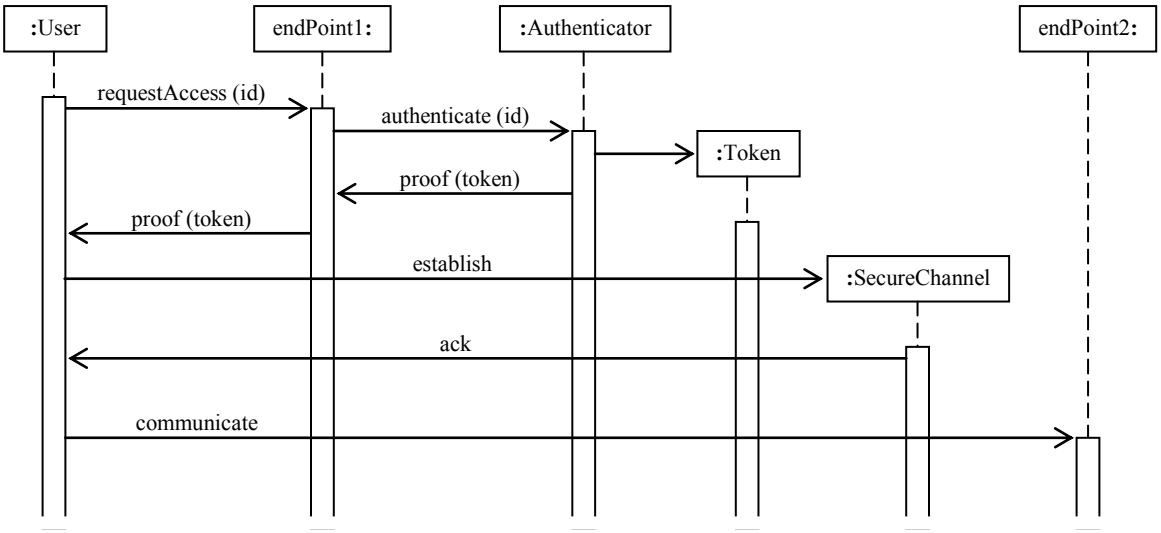


Figure 6.4 Sequence diagram for accessing an endpoint.

Implementation

Define first the end points which the VPN will reach. Consider the architectural layer where the communications should be secure according to the needs of the applications. After this decision, use the concrete VPN at the corresponding level, IP or TCP. See the corresponding patterns (IPSec VPN and TLS VPN) for help to make this decision. Both sides must share a public key system for authentication and must have appropriate software packages running in them.

Example Resolved

Now the users can be authenticated at the end points, which assures that they are talking with those they want to talk. User messages are now protected from external attacks when going over the secure channel.

Variants or Concrete patterns

Virtual Private Networks can be established at the Application layer (XML or Application VPN), TLS (SSL) VPNs are established at the transport layer. IPSec VPNs are established at the IP layer. Because of their importance, we describe the last two below as separate patterns.

Known Uses

- Citrix provides a site to site SSL VPN connection for remote users to log into the secure network as well as access applications on the company (secure) network [Cit].
- Cisco VPN uses an IPSec VPN and they provide authorization [Cis].
- Nokia provides a VPN connection for Nokia Mobile Users.

Consequences

This pattern has the following advantages:

- We can use the Internet or other insecure networks to reduce the cost.
- Cryptography can protect our messages from being read or modified by attackers.
- Authentication at end points assures that only registered users can access the secure channel.
- Mutual authentication between end users is possible.
- The system can accommodate new links for new users by just replicating the access software.
- We can use any cryptographic algorithm to establish the secure channel, which allows us to make tradeoffs between security and cost.

- We can add authorization to access specific resources at each end point.
- We can add a logging system for the users logging in at the end points for future audits.
- The VPN is transparent to the users, who are authenticated by their local endpoints. The VPN is a client-server architecture easy to configure.
- We can have different versions of the pattern that can use the specific features of each protocol layer.

This pattern has the following disadvantages:

- If the VPN connection is compromised, the attacker could get full access to the internal network. Authorization can restrict this access, however.
- Because of encryption, VPN traffic is invisible to IDS monitoring. If the IDS probe is outside the VPN server, as is often the case, then the IDS cannot see the traffic within the VPN tunnel. Therefore if a hacker gains access to the end node of the VPN, he can attack the internal systems without being detected by the IDS.
- In case of VPN with a private end user, the remote computer used by the private user is vulnerable to outside attacks which in turn can attack the network to which it is connected.
- There is some overhead in the encryption process.

Related Patterns

- Firewalls can be added to each endpoint to filter inputs [Sch06]. They can protect against some types of attacks coming from untrusted sources.

- IDS can be added in each of the network layers to detect attacks in real time [Fer05, Kum12a].
- The VPN uses the Secure Channel pattern that in turn uses cryptography to protect its messages [Bra00].
- The Authenticator [Sch06] can authenticate users and nodes.
- Access Control/Authorization can be added in each site to control access to specific resources [Sch06].

6.2 IPSec VPN

This pattern sets up a secure channel between two endpoints using cryptographic tunneling through the IP layer, with authentication at each end point.

Context

Users scattered in many predefined locations, who need to communicate securely with each other, using the Internet or another insecure network.

Problem

Assuming that we need to communicate using the IP protocol, how do we establish a secure channel for the end users of a network so they can exchange messages through some fixed points?

The solution is affected by the following forces:

- The number and required speed of the communications must decide the type of protection we use. The use of IPSec would provide higher speed between fixed physical locations [Sta03].

- Communication at the IP level includes the network, servers, and routers. Messages should be protected going through all of them.

Solution

Implement the cryptographic tunnel at the IP level using the facilities of IPSec.

Structure

The class diagram for the IPSec VPN pattern is similar to the one in Figure 6.3 which describes pattern for abstract VPN.

Implementation

Designing the architecture of the IPSec protocol includes appropriate host placement (for host-to-host architectures) and/or gateway placement (for host-to-gateway and gateway-to-gateway architectures) [Wei02]. Both sides must share a public key system for authentication and must have appropriate software packages running in them.

The packet filter firewall determines which types of traffic should be permitted and denied, and what protection and compression measures (if any) should be applied to each type of permitted traffic (e.g., ESP tunnel using AES for encryption and HMAC-SHA-1 for integrity protection; LZS for compression). Encapsulating Security Payload (ESP), is a sub-protocol of IPSec that provides confidentiality, data origin authentication,

integrity, and replay protection [Sta03] as seen in Chapter 5. AES is the Advanced Encryption Standard [For04]. HMAC stands for Hash-based Message Authentication Code, and SHA-1is a specific hash algorithm [For04, Sta03]; they are used for protecting message integrity.

Variants

We can add authorization for the end users.

Known Uses

- Cisco has an IPSec VPN and they also provide authorization [Cis].
- Cyberoam offers an identity-based IPSec VPN [Cyb].
- Check Point's VPN Software Blade is an IPSec VPN that integrates access control, authentication and encryption [Che].

Consequences

This pattern has the following advantages:

- IPSec is supported by most operating systems
- The VPN is transparent to clients in gateway-to-gateway architectures.
- We can use a variety of authentication protocols.

This pattern has the following disadvantages:

- Can only protect IP-based communications.
- Requires client software to be configured (and installed on hosts without a built-in client) for host-to-gateway and host-to-host architectures.

- Does not protect communications between the clients and the IPSec gateway in gateway-to-gateway architectures.
- IPSec VPNs require large software packages, typically 6-8 MB, and may be difficult to configure.

Related Patterns

- Firewalls can be added to each endpoint to filter inputs [Sch06]. They can protect against some types of attacks coming from untrusted sources.
- IDS can be added in each of the network layers to detect attacks in real time [Fer05, Kum12a].
- The VPN uses the Secure Channel pattern that in turn uses cryptography to protect its messages [Bra00].
- The Authenticator [Sch06] can authenticate users and nodes.
- Access Control/Authorization can be added in each site to control access to specific resources [Sch06].

6.3 TLS (SSL) VPN

This pattern sets up a secure channel between two endpoints using cryptographic tunneling through the transport layer, with authentication and access control (authorization) at each end point

Example

Our company has a web-based eCommerce site. We need to assure the customers that they are interacting with the proper application and that they can send their credit information to buy items in a secure way.

Context

A large number of users scattered in many locations, need to communicate securely with each other, using the Internet or another insecure network. Most of the interactions occur through web sites.

Problem

How do we establish a secure channel through the transport layer for the end users of a network so they can exchange messages through some fixed points?

The solution is affected by the following forces:

- Messages will go from a process to another process through servers and routers. The message should keep its security in this communication.
- The performance should be good at normal and peak loads.

Solution

Use TLS reverse proxy servers (commonly referred to as *SSL proxy servers*) to connect remote users. A remote user who needs to access some of the organization's applications uses the main Uniform Resource Locator (URL) for the proxy server in a web browser and connects to it through TLS-protected HTTP. The user then authenticates to the proxy server. Once authenticated, the user can access designated applications, as specified in the proxy server's access controls.

Structure

Figure 6.5 shows the class diagram for the TLS VPN pattern. A **Proxy** represents the endpoint and has the functions of **Authentication**, **Secure Channel**, and **Access Control** (Authorization).

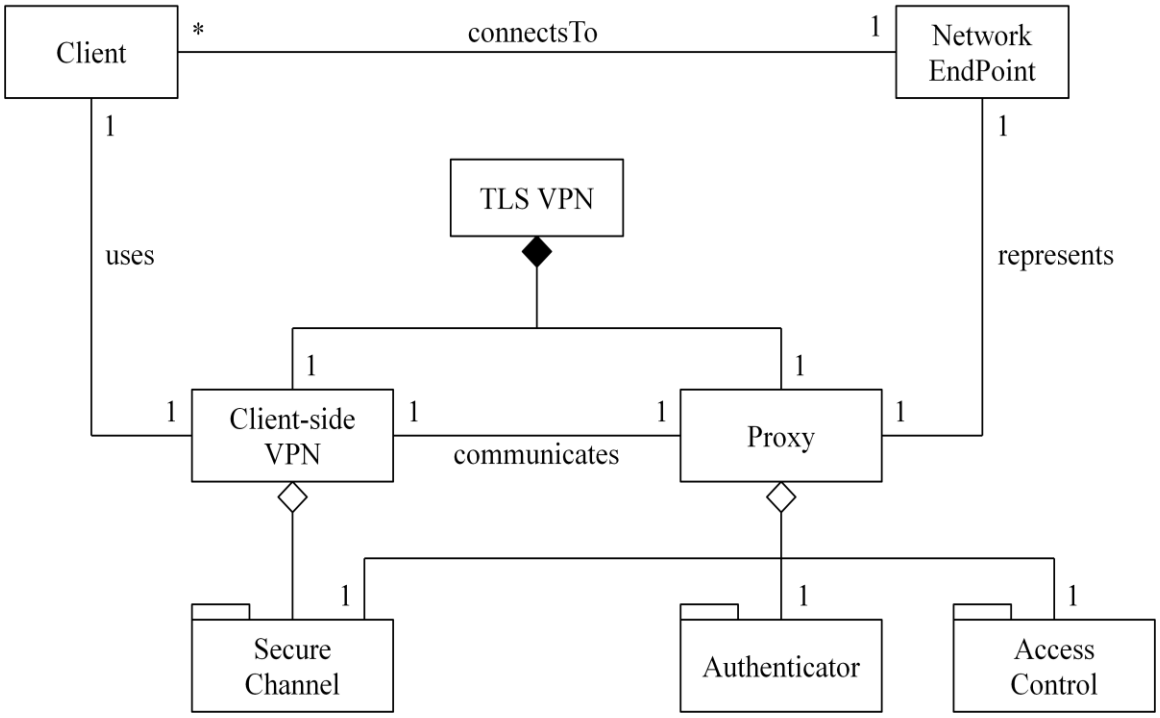


Figure 6.5 Class diagram for Security Pattern for TLS VPN.

Implementation

An authentication algorithm is implemented at the server which authenticates the server to the client [Hey07]. A secure channel is established over the public network using a suitable cryptographic algorithm which allows users to communicate securely

with the servers. Some TLS VPNs provide hardware accelerators. Both the client and the server must have preloaded VPN software.

Example Resolved

The SSL proxy server can authenticate the server to the user and establishes a secure channel so that the remote users can send their credit information in encrypted form, thus protecting it from eavesdropping attacks.

Known Uses

- Citrix provides a site to site SSL VPN connection for remote users to log into the secure network as well as access applications on the company (secure) network [Cit].
- SonicWALL acquired Aventail and its TLS VPN. This product includes authentication and network access control [Sonic].
- Cyberoam has an identity-based TLS VPN [Cyb].
- Aventail, Cisco, Juniper, Microsoft, and Nokia also provide TLS VPNs.

Consequences

This pattern has the following advantages:

- If access is needed to only Web-based applications, the solution is very convenient for users and easier to deploy and maintain than remote access solutions that involve client installation or configuration.
- The proxy server can authenticate users before they can gain any access to applications, as opposed to allowing users to connect directly to individual

applications' login screens. This adds another layer of security by only allowing authenticated users to see what applications are being served.

- Since the client systems connect above the network layer, they are not on the network in the same manner that IPsec client systems would be. This severely reduces their ability to attack or misuse systems on the organization's networks.
- The proxy server can authenticate itself to the user by means of a certificate.
- Logging is now more convenient, it is just another function of the proxy.

This pattern has the following disadvantages:

- Non-web-based applications and applications that are more challenging to proxy (e.g., those that use multiple dynamic ports) typically require additional software and services, such as terminal servers and special client software. This makes the solution more resource-intensive to deploy and less convenient to use.
- A compromise of the proxy server could allow an attacker to intercept data and authentication credentials for many different applications at once.
- TLS (SSL) is a complex protocol, which has been found to have security problems in some implementations. This means that the degree of security reachable with this pattern may not be as high as with the IPSec VPN.

Related Patterns

- Firewalls can be added to each endpoint to filter inputs [Sch06]. They can protect against some types of attacks coming from untrusted sources.
- IDS can be added in each of the network layers to detect attacks in real time [Fer05].
- The VPN uses the Secure Channel pattern that in turn uses cryptography to protect its messages [Bra00].
- The Authenticator [Sch06] can authenticate users and nodes.
- Access Control/Authorization can be added in each site to control access to specific resources [Sch06].
- Proxy is a pattern in [Gam94]. In this case it intercepts requests going to the endpoints and performs the required checks.

Conclusions

A virtual private network is a useful component in network architectures. We presented here an abstract pattern for the VPN architecture and patterns for the IPSec VPN and for the TLS VPN.

7 QUALITATIVE ANALYSIS OF NETWORK PATTERNS

In this chapter we analyze the patterns developed by us and try to evaluate the patterns based on three sets of distinct criteria.

We have seen that each of these security patterns is approached as a solution to resolve certain forces. For example we know that each of these patterns was developed for handling security threats and should be adaptable, effective, easily deployable, easy to use, etc.. So firstly we try to define these common forces applicable to any new security pattern and use them as a yardstick to do a qualitative comparison. These criteria and the comparison results are given in next section 7.1.

The second set of criteria can be described as how well a specific security pattern might respond to different categories of attacks as they are described by [How02]. To describe the different categories of attacks that are possible in a software system Howard and LeBlanc [How02] propose the STRIDE model. Here we evaluate the security patterns we developed in Chapters 4, 5 and 6 based on how well a software system using a specific security pattern might respond to each category of possible attack. The categories of attack based on the STRIDE model are briefly described in section 7.2.

As third point we apply an approach using a matrix defined by dividing the problem space along multiple dimensions, and allowing patterns to occupy regions,

defined by multiple cells in the matrix [Van09]. This classification of security patterns, which is based on the needs of the users, is presented in section 7.3.

7.1 Evaluation of the Patterns based on their Forces.

The first evaluation of the patterns is based on the following criteria which usually are part of the forces for each of these patterns:

Adaptability:

A pattern is said to be adaptable if it can accommodate changing threats. For example a TLS protocol pattern should be able to handle all attacks on the application messages communicated over the transport layer of the internet.

Auditability:

A pattern is said to be auditable if the pattern has the provision to keep an audit trail. For example if we have all the activities logged, when an external attack happens, in an audit trail we may be able to analyze the events that caused the attack and find evidence of the attack as well as get tips on how to improve the system. The audit trail may not be defined as part of the pattern itself but may be part of a shared system facility. What matters is the possibility of collecting useful information in the classes of the pattern.

Effectiveness:

A pattern is effective if the infrastructure provided by the pattern is able to handle appropriately a type of threat. For example, a security pattern such as an IDS

pattern should detect a large variety of intrusions and respond appropriately to all of them.

Complexity:

If a pattern is not complex it can be easily managed. On the other hand if we use a very complex security pattern, the administrators using the pattern in a real time environment may have a tough time configuring the system, which may lead to security vulnerabilities.

Usability:

The pattern must present a clear security picture to administrators, or they will make mistakes. Usability measures the ease with which the unit of a security pattern can be configured and used in an actual scenario.

Overhead:

We can estimate the degree of overhead of a pattern. For example, a security pattern might include algorithms that take a long time or need a long sequence of operations or need to exchange many messages. There may be two different patterns or variants of the same pattern that have different overheads and depending on the application one pattern is to be preferred to the other.

Cost:

We can analyze the potential cost of implementing a security pattern. Different patterns or variants of the same pattern may have different costs depending on

deployment needs or the need for specialized support. Depending on the application, we can select the one with lower cost although it may not be as secure as another.

Deployability:

We can estimate the ease with which the units of these patterns can be deployed in an actual system. This aspect has a direct effect on the overhead.

Modifiability/Extensibility:

A pattern is modifiable or extensible if it can be easily modified or extended to handle new threats or functions or variations of its functions

Scalability:

A pattern is scalable if the pattern can conveniently include more users, more entries, or more stake holders. For example, if a company adopts a security pattern for its day to day email security, the units of the pattern should be able to handle the case when the employees or the email traffic double in a short time.

In Table 7.1, the details of the evaluation of the security patterns are shown based on the above described forces. We have used a 3 - point scale where 1, 2 and 3 denote that the pattern satisfies the force in a low, medium and high manner respectively. For example a signature based IDS has a low adaptability (1) against new threats, but has a high value (3) of auditability.

Table 7.1 Evaluation of the Security Patterns based on their Forces.

Pattern	Adaptability	Auditability	Effectiveness	Complexity	Usability	Overhead	Cost	Deployability	Modifiability/Extensibility	Scalabbility
Abstract IDS										
Signature Based IDS	**1**	**3**	**2**	**2**	**3**	**3**	**2**	**3**	**1**	**1**
Behavior Based IDS	**2**	**3**	**2**	**1**	**2**	**2**	**1**	**2**	**3**	**1**
Abstract Network Protocol										
IPSec Protocol	**3**	**1**	**3**	**3**	**3**	**2**	**2**	**2**	**3**	**3**
TLS Protocol	**2**	**3**	**3**	**3**	**3**	**2**	**2**	**2**	**3**	**3**
Abstract VPN										
IPSec VPN	**3**	**1**	**3**	**3**	**3**	**2**	**2**	**2**	**3**	**1**
TLS VPN	**2**	**3**	**3**	**3**	**3**	**2**	**2**	**2**	**3**	**1**

Legend: **1** – Satisfies Low; **2** – Satisfies Medium; **3** – Satisfies High

7.2 Evaluation of the Patterns based on STRIDE Model

In this evaluation the criteria can be described as how well a specific security pattern might respond to different categories of attacks as they are described by [How02]. To describe the different categories of attacks that are possible in a software system Howard and LeBlanc propose the *STRIDE* model.

- **S:** The first category of attacks consists of the *Spoofing identity (S)* attacks. Identity spoofing is illegally acquiring and then using another user's authentication information.
- **T:** The second category of attacks consists of the *Tampering with data (T)* attacks. Data tampering involves malicious modification of data.
- **R:** The third category of attacks consists of the Repudiation attacks. *Repudiation (R)* attacks are associated with users who deny performing an action without other parties having a way to prove otherwise.
- **I:** The fourth category of attacks consists of the *Information disclosure (I)* attacks. Information disclosure threats involve the exposure of information to individuals who are not supposed to have access to it.
- **D:** The fifth category of attacks consists of *Denial of Service (D)* attacks. Denial of Service (DoS) attacks deny service to valid users.
- **E:** Finally, the sixth category of attacks consists of the *Elevation of privilege (E)* attacks. In this type of attack, an unprivileged user gains privileged access and therefore has sufficient access to compromise or destroy the entire system, if only one level of privilege is used [Hal06].

Table 7.2 gives the evaluation parameters for the security patterns developed by us using the STRIDE model to get the categories of attacks. But the evaluation parameters have been defined by us and are D – whether the security pattern can Detect an attack, P – whether the security pattern can offer Protection against the category of attack and E - whether the security pattern can offer Enhanced or extra protection.

Table 7.2 Evaluation of the Security Patterns based on the STRIDE model.

Pattern	S	T	R	I	D	E
Abstract IDS	D				D	
Signature Based IDS	D				D	
Behavior Based IDS	D				D	
Abstract Network Protocol	P	P	P	P	P	
IPSec Protocol	E	P	E	E	P	
TLS Protocol	P	P	E	E	P	
Abstract VPN	P	P	P	P	P	
IPSec VPN	E	P	E	E	P	
TLS VPN	P	P	E	E	P	

Explanations: **D**- Detection of threat exists; **P** – Protection against threat exists;

E– Enhanced protection exists

7.3 Evaluations Based on Pseudo Hypercube Matrix

Most security pattern classifications are typically hierarchical. [Van09] has proposed to address pattern classification and problem coverage through the use of a multi-dimensional matrix of concerns. Each dimension of the matrix is a distinct list of concerns along a single axis, with a simple concept and a set of distinctions that define the categories. The categories along an axis or dimension should be easily understood and represent widely used and accepted classifications with respect to that concept.

The six dimensions of classification are shown graphically in Figure 7.1. The "all" classification is used when the distinctions along that axis are not meaningful to a pattern. The six dimensions are *lifecycle stage*, *component (code)* source, *response*, *architectural layer*, *constraint* and finally *domain*. Brief explanations of these dimensions are given below [Van09].

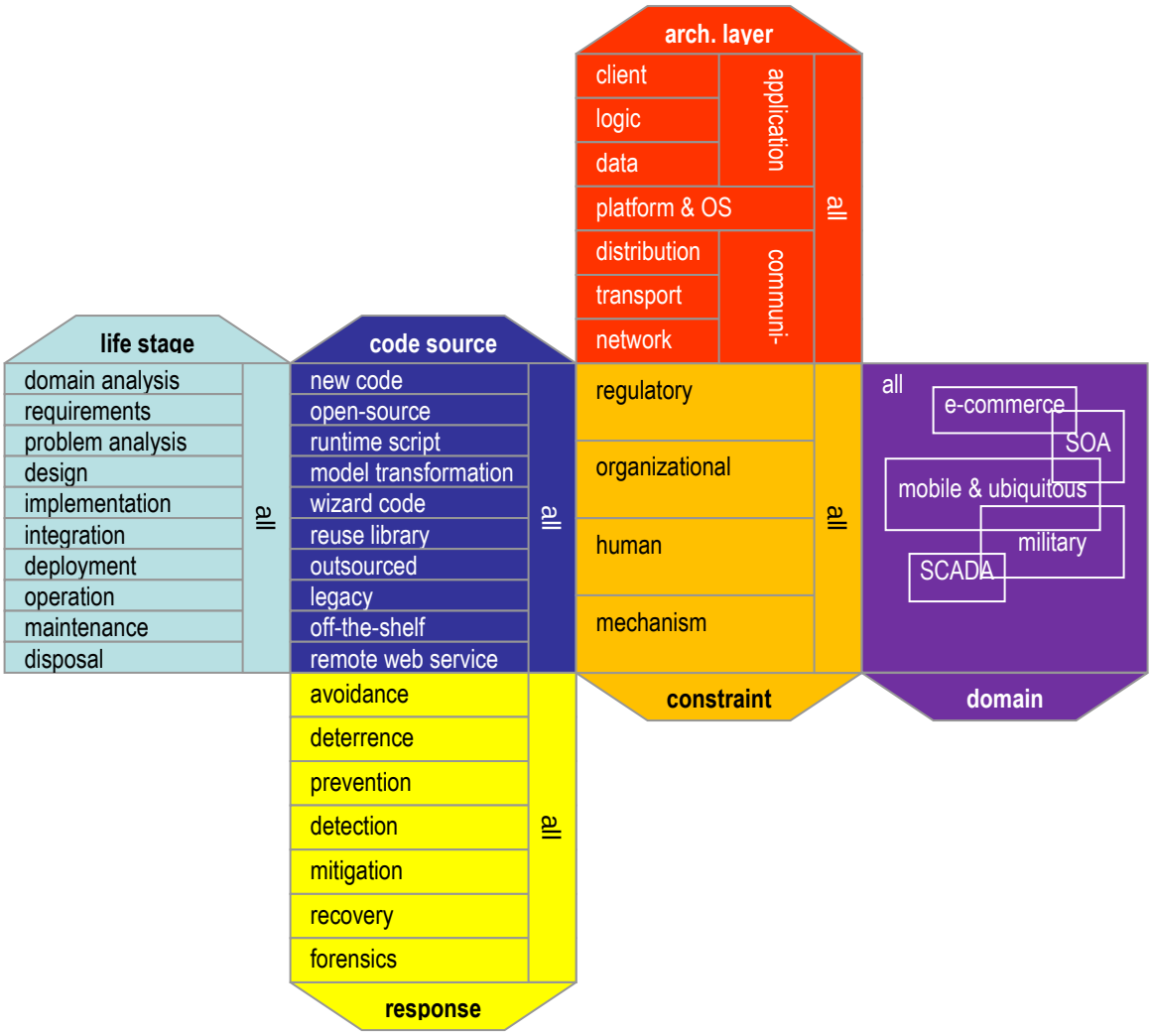


Figure 7.1 Six primary dimensions of classification of Security Patterns (shown as a pseudo-hypercube) [from Van09].

The first dimension would be a list of life-cycle activities, covering domain analysis, requirements, problem analysis, design, implementation, integration, deployment (including configuration), operation, maintenance and disposal. It actually starts with a pre-beginning or potential, where domain analysis is performed, and could end with disposal.

The categories of component source are organized around a dichotomy of internal vs. external control. In new code, the developer has complete control over all details. With an outsourcing service, there is little or no control or even knowledge of details, and limited ability to test.

The response axis is based on whether or not an attack happens and it goes from not happening at all (avoidance), to incidence (happened) and to already happened (forensics).

Architectural layers provide another useful dimension, since problems and their solutions in different layers of the architecture differ. Roughly the same architecture continuum has been divided in different ways for communication protocols, business systems, and execution environments, but always with an ordering from low to high level of abstraction, and from network to platform to application. We fit these concepts on an axis from meaningless bits to end-task semantics. Combining elements from several domains, and allowing for overlap between views, we chose the following distinctions: network, transport, distribution (including gateways and brokers), platform and operating system, data, business logic, and client. A simpler notion of application spans the last three. Network, transport, and distribution may also be grouped as communication. Since patterns can be placed in more than one cell, there is no need for disjoint classification.

Leveson defines four levels of constraint: mechanism, human (operator or developer), organizational, and regulatory. In Leveson's work on system safety [Lev04], each level of constraint plays an important role in safety failures and their prevention. By

extension, we use the same levels for security with an axis with levels from thing to society. While most security patterns describe mechanisms, some security patterns are mostly concerned with practices, policies, and regulations [Van09].

Some solutions are specific to a particular domain or application type. Ubiquitous computing, e-commerce, and SCADA (Supervisory Control and Data Acquisition) all pose distinct security challenges. For example, security for 3-Tier business applications may differ from solutions for SCADA systems of sensors and controls. This axis is an exception in that it does not have a dichotomy or ordering – the space is freely defined. Organizations can create patterns for their own domain as a form of knowledge capture.

Now let us apply the matrix to the security patterns we developed as part of this dissertation.

Figure 7.2 shows the classifications of the abstract IDS pattern. The pattern describes a mechanism that is applied in the design stage of development. The type of response is detection of the attack before it happens. The architectural layer used is all the communication layers. Component source is not relevant as it is applicable to all classifications along this axis. The pattern is specific to all domains and IDS usually complements a firewall in most of the domains.

The same classification, as shown in Figure 7.2, can be extended to the *Rule - based* and *Signature – based IDS* security patterns we wrote.

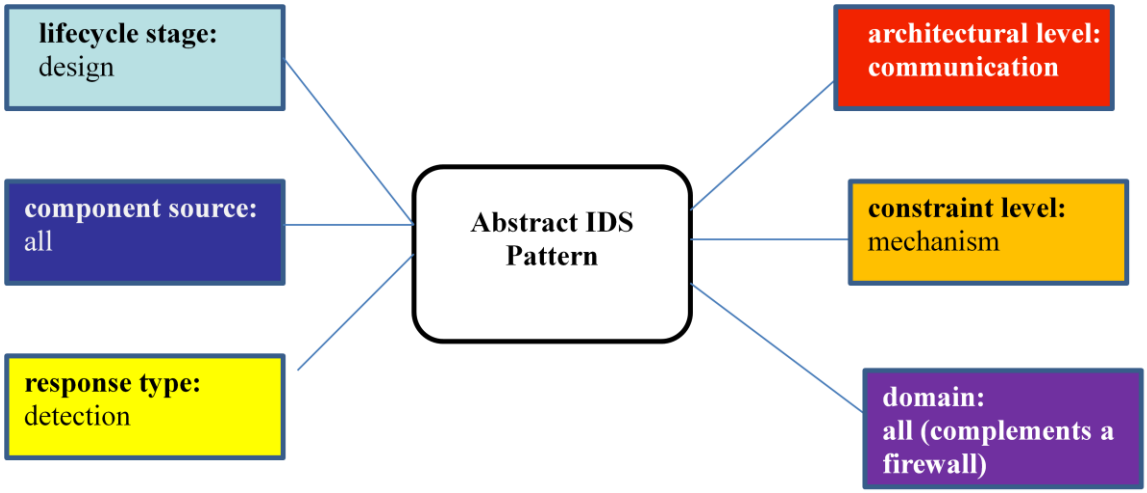


Figure 7.2 Classifications of the abstract IDS pattern.

Figure 7.3 shows the classifications of the IPSec protocol pattern. The pattern describes a mechanism that is applied in the design stage of development. The type of response is prevention of the attack. The architectural layer where this pattern is applied is the network layer. The pattern is specific to any domain that uses IP packets for data communication.

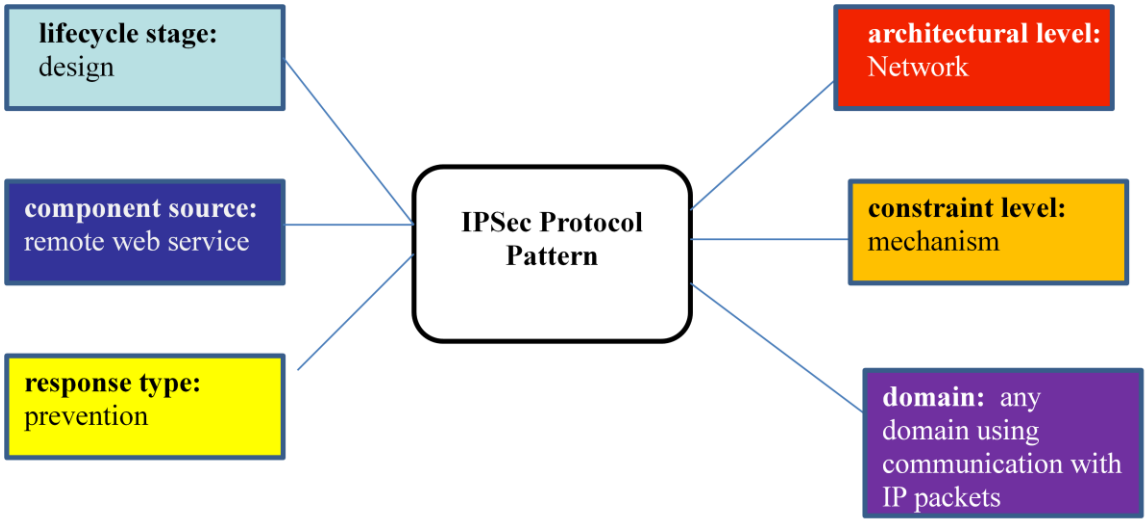


Figure 7.3 Classifications of the IPSec protocol pattern.

Figure 7.4 shows the classifications of the TLS protocol pattern. This pattern also describes a mechanism that is applied in the design stage of development. The type of response is prevention of the attack. The architectural layer where this pattern is applied is the transport layer. The pattern is specific to any domain that uses the transport layer for data communication.

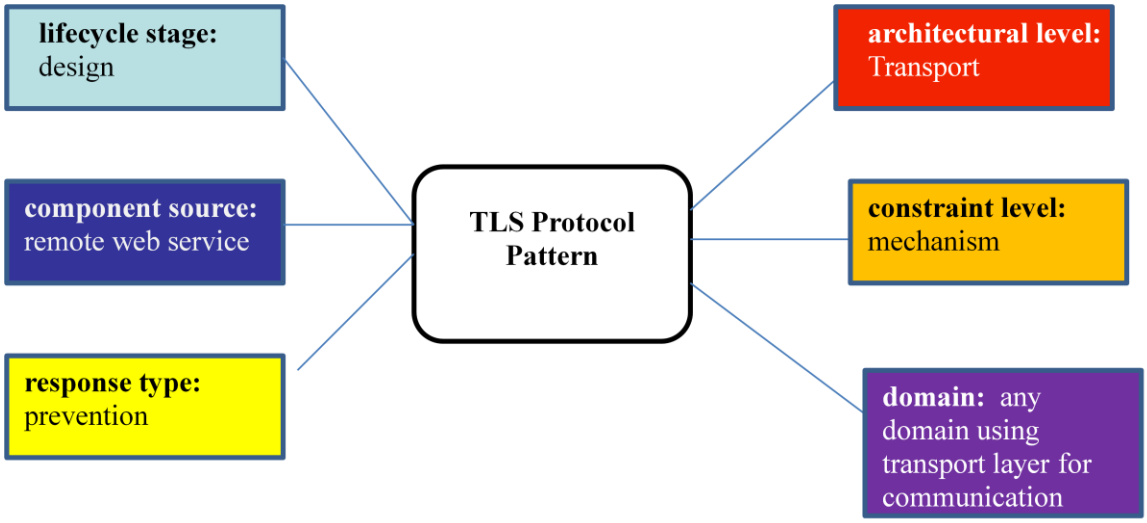


Figure 7.4 Classifications of the TLS protocol pattern.

Figure 7.5 shows the classifications of the abstract VPN pattern. This pattern also describes a mechanism that is applied in the design stage of development. The type of response is prevention of the attack. The architectural layer where this pattern is applied is the communication layers. The pattern is specific to all domains where users need to connect to the internet, mutually authenticate and send data through a secure channel.

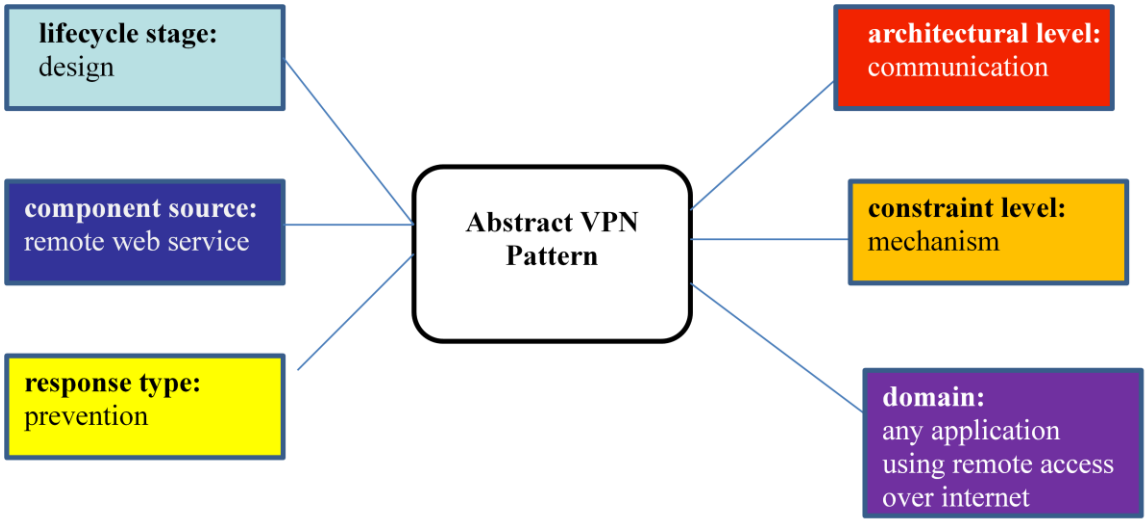


Figure 7.5 Classifications of the abstract VPN pattern.

The *IPSec VPN pattern* and *TLS VPN pattern* will be identical to the abstract VPN pattern shown in Figure 7.5 except for the architectural layer. For IPSec pattern the architectural layer will be the network layer and for TLS, it will be the transport layer respectively.

The qualitative evaluation presented in this chapter can help in selecting security patterns in order to build a secure software system that can offer a reasonable protection against the most common attacks. As an example, Supervisory Control and Data Acquisition (SCADA) systems consist of geographically scattered units (*field devices*) controlled using centralized data acquisition and control (*control center*) [Sto06]. Section 16.5 in the [Fer13] book explains how to make a SCADA system secure, using a combination of different security patterns. In this example we can see how to handle Denial of Service (DoS) threats by applying the *firewall pattern* with the ***IDS pattern***

which we developed as part of this thesis. This example also illustrates how threats such as sniffing of data and identity spoofing can be handled by using ***VPN pattern,*** also developed as part of this dissertation, which includes the *authenticator pattern* and the *secure channel pattern*.

But we believe that with the qualitative evaluation of the security patterns, a quantitative approach, in evaluating the security of software systems, is also desirable to convince ourselves that the system is secure. For example, we can combine software metrics' techniques with the use of security patterns so that software designs could be quantitatively evaluated in terms of security [Hal06].

8 CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

This work gives an overview of the major internet network layers, the security patterns for different security protocols associated with these layers and the security patterns for the corresponding security components such as firewalls, IDS and VPN.

We have emphasized the importance of patterns and security patterns in Chapter 2, in particular to help designers and users build and understand complex systems such as the Internet.

We did a reconnaissance survey (Chapter 3) of available security patterns on network transmission, the boundary security devices and the associated security protocols. In the same chapter we identified the patterns that were missing which formed the basis of the patterns developed in Chapters 4, 5 and 6. We also discussed briefly in the same chapter patterns for specialized networks such as VoIP and wireless networks.

We have developed an abstract pattern for IDS (Chapter 4). From the abstract pattern we derived two types of concrete IDS patterns: (Chapter 4)

- Anomaly or Behavior Based IDS
- Rule or Signature Based IDS.

In the same chapter we have explained how an IDS complements a firewall and adds extra security to the network.

We have presented an abstract pattern for network layer protocols (Chapter 5). From this abstract pattern we developed concrete patterns for the IPSec protocol used at the network layer and the TLS protocol used at the transport layer (Chapter 5).

In Chapter 6 we have described the abstract VPN pattern. The abstract VPN pattern is made concrete when we define patterns for TLS VPN and IPSec VPN.

We investigate the qualitative features of the security patterns developed by us, by evaluating each pattern based on three sets of unique criteria in Chapter 7.

The major contribution of this dissertation is that, we have tried to unify the security of the network layers using security patterns by tying in security patterns for network transmission, network protocols and network boundary devices.

We analyzed the main use cases generated by these patterns and enumerated the possible threats in these scenarios. Thus these patterns provide a good infrastructure for software engineers in designing a secure network.

The qualitative analysis we did in Chapter 7 based on the three different criteria – firstly the forces, secondly the response to different categories of attack and finally the pattern classification and problem coverage through the use of a multi-dimensional matrix of concerns, helps software engineers to select security patterns for their system based on the forces they need to resolve, the response they need for the common

categories of attack they face or might face and their concerns based on the multi-dimensional matrix of concerns.

8.2 Future Work

Scope for future work is unlimited. I have mentioned a few possible areas where future work is possible:

- **Develop patterns at the application layer:**

  As you have seen in the thesis the patterns have been limited to the network layer and the transport layer. Find the missing patterns at the application layer for the boundary devices and protocols.

  - We have developed patterns for IPSec VPN and TLS VPN. This can be extended to the network application layer and a security pattern for WSS VPN can be developed on the Web Services Standards (WSS) protocol.
  - Similarly we can develop a pattern for an application layer IDS.

Table 8.1 tabulates already existing patterns (in Italics), patterns written as part of this thesis (in Bold) and patterns at the application layer to be developed (in Bold and Italics).

Table 8.1 Network Layers and Patterns

	Firewall	**IDS**	**VPN**	**Secure Protocol**
User Application with webservices	*XML FW*	***XML IDS***	***XML VPN***	***SAML***
TCP	*Proxy FW*	**TCP IDS**	**TLS/SSL VPN**	**TLS**
IP	*Packet filter*	**Packet IDS**	**IPSec VPN**	**IPSec**

Legend: *Italics* – Already Existing Patterns

Bold – Patterns written as part of this thesis.

Italics and Bold – Patterns yet to be developed

- **Validation of the patterns by using them in an application:**

  A way to validate the proposed patterns is to apply them to a real application.

  - Supervisory Control and Data Acquisition (SCADA) systems consist of geographically scattered units (*field devices*) controlled using centralized data acquisition and control (*control center*) [Sto06]. A power plant is an example of a SCADA system. We can apply our model to a SCADA system and

compare the results to other analysis of SCADA security such as [Fer10, Igu06].

- **Catalog of Patterns:**
    - [Fer13] has a catalog of around 70 patterns. We can create a separate catalog for network-related patterns with more use cases which can be used for engineers and software designers when they need to design a secure network.
- **Specialized Network Patterns**:
    - In our survey we mentioned specialized networks such as VoIP networks and wireless networks. Patterns for VoIP networks are available in [Fer07]. We can find the missing patterns for a wireless network and write them. For example we can develop a TLS Protocol pattern catered for a wireless network based on the TLS protocol security pattern we developed in Chapter 5.
- We can identify other specialized network patterns and try to identify the missing security patterns and try to develop them. For example as we mentioned above SCADA network is one of the networks, where we can try to modify or enhance the existing patterns and develop new patterns.

REFERENCES

[Air] Airtight Networks: Wireless Intrusion Prevention:

http://www.airtightnetworks.com/home/solutions/wireless-intrusion prevention.html. Last Accessed on December, 21, 2013.

[Bad04] M.Badra, A.Serhrouchni and P.Urien. "A lightweight identity authentication protocol for wireless networks", *Computer Communications*, Volume 27, Issue 17, 1 November 2004.

[Bad09] M.Badra and Hajjeh, Internet-Draft, (D) "TLS Multiplexing", April 2009

http://tools.ietf.org/html/draft-badra-hajjeh-mtls-05. Last Accessed on December, 21, 2013.

[Bie01] E.Biermann, E.Cloete and L.M.Venter. "A comparison of Intrusion Detection systems", *Computers & Security*, Volume 20, Issue 8, December 2001, Pages 676-683.

[Bra98] A.Braga, C.Rubira, and R.Dahab, "Tropyc: A pattern language for cryptographic object-oriented software". Chapter 16 in *Pattern Languages of Program Design 4* (N. Harrison, B. Foote, and H. Rohnert, Eds.). 1998.

[Bro99] F.L.Brown, J.DeVietri, G.Diaz de Villegas and E.B.Fernandez, "The Authenticator pattern", *Proceedingss. of the Pattern Languages of Programs Conference (PLoP)*, 1999. http://hillside.net/plop/1999/

[Bus96] F.Buschmann, R.Meunier, H.Rohner., P.Sommerlad, and M.Stal (1996). "*Pattern oriented software architecture: A system of patterns*". West Sussex, England: Wiley.

[Che] Checkpoint Software Technologies Ltd:, Checkpoint IPS-1: http://www.checkpoint.com/products/ips-1/index.html. Last Accessed on December, 21, 2013

[Cis] Cisco VPN:Client http://www.cisco.com/c/en/us/products/security/vpn-client/index.html/, Last Accessed on December, 21, 2013

[Cis00] Cisco Systems: Products and Technologies > Cisco Intrusion Detection:

http://www.cisco.com/c/en/us/products/security/ips-4200-series sensors/index.html. Last Accessed on December, 21, 2013

[Cis01] "The Internet Protocol Journal - Volume 3, No. 1" March 2000

http://www.cisco.com/web/about/ac123/ac147/ac174/ac197/about_cisco_ipj_archive_article09186a00800c830b.html. Last Accessed on December, 21, 2013

[Cit] Citrix: "Using an SSL VPN to provide secure remote access" http://www.citrix.com/products/netscaler-gateway/resources-and-support/ssl-vpn.html. Last Accessed on December, 21, 2013

[Cyb] Cyberoam VPN: http://www.cyberoam.com/vpn.html. Last Accessed on December, 21, 2013

[Das98] F.Das Neves and A.Garrido, "Bodyguard", Chapter 13 in *Pattern Languages of Program Design 3*, Addison-Wesley, 1998.

[De01] J.T.De Souza and S.Matwin. "A pattern language for providing client-server confidential communication". In: *Proceedings of SugarLoafPLoP 2001*. Rio de Janeiro, Brazil; 2001.

[Del04] N.Delessy, E.B.Fernandez, S. Rajput, and M.M Larrondo-Petrie, "Patterns for application firewalls", *Proceedings of the Pattern Languages of Programs Conference (PLoP)* 2004, http://hillside.net/plop/2004/

[Del07] N.Delessy, E.B.Fernandez, M.M.Larrondo-Petrie and J.Wu. “Patterns for access control in distributed systems”. In: *Proceedings of the 14th Pattern Languages of Programs conference (PLoP2007)*. Monticello, Illinois, USA; 2007

[Dep05] O.Depren, M.Topallar, E.Anarim and M.K.Ciliz, “An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks”, *Expert Systems with Applications*, Volume 29, Issue 4, November 2005, Pages 713–722

[Elg06] A.Elgohary, T.S.Sobh and M.Zaki. “Design of an enhancement for SSL/TLS protocols” *Computers & Security* Volume 25, Issue 4, June 2006.

[Fer93] E.B.Fernandez, M.M.Larrondo-Petrie and E.Gudes “A method-based authorization model for object-oriented databases”. In: *Proceedings of the OOPSLA 1993 conference workshop on security in object-oriented systems;* 1993. p. 70–79

[Fer01] E.B.Fernandez and R.Pan “A pattern language for security models”, *Proceedings of the Pattern Languages of Programs Conference (PLoP)*, 2001. http://hillside.net/plop/plop2001/

[Fer02] E.B.Fernandez “Patterns for operating systems access control”. *Proceedings of Pattern Languages of Programs Conference (PLoP)* 2002. http://hillside.net/plop/plop2002/

[Fer03a] E.B.Fernandez and R.Warrier, “Remote authenticator/authorizer”. In: *Proceedings of the 10th conference on Pattern Languages of Programs (PLoP 2003)*; 2003. http://hillside.net/plop/plop2003/

[Fer03b] E.B.Fernandez, M.M.Larrondo-Petrie, N.Seliya, and A.Herzberg. "A Pattern language for firewalls". *In Proceedings of the Pattern Languages of Programs (PLoP) Conference, 2003*. http://hillside.net/plop/plop2003/

[Fer05] E.B.Fernandez and A.Kumar, "A security pattern for rule-based intrusion detection", *Proceedings of the Nordic Conference on Pattern Languages of Programs, Viking PLoP 2005*, Otaniemi, Finland, 23-25 September 2005.

[Fer07] E.B.Fernandez, J.C.Pelaez, and M.M.Larrondo-Petrie, "Security patterns for voice over IP networks", *Journal of Software*, Vol. 2, No 2, August 2007, P19-29

[Fer08] E.B.Fernandez, H.Washizaki, and N.Yoshioka, "Abstract security patterns", Position paper in *Procs. of the 2nd Workshop on Software Patterns and Quality* (SPAQu'08), in conjuction with the 15th Conf. on Pattern Languages of Programs (PLoP 2008), October 18-20, Nashville,TN.http://patternswg.fuka.info.waseda.ac.jp/SPAQU/index.html Or http://hillside.net/plop/2008/papers/ACMVersions/spaqu/fernandez.pdf

[Fer10] E.B.Fernandez and M.M.Larrondo-Petrie, "Designing secure SCADA systems using security patterns", *Proceedings of the 43rd Hawaii Conf. on Systems Science*, Honolulu, HI, Jan.2010, P1-8. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5428672

[Fer10b] N.Ferguson, B.Schneier and T.Kohno." *Cryptography engineering: design principles and practical applications*" Wiley (2010)

[Fer12] E.B.Fernandez, O.Ajaj, I.Buckley, N.Delessy-Gassant, K.Hashizume, and M.M.Larrondo-Petrie, "A Survey of Patterns for Web Services Security and Reliability Standards", *Future Internet* 2012, 4(2), 430-450. http://www.mdpi.com/1999-5903/4/2/430.

[Fer13] E.B.Fernandez, "*Security patterns in practice: Building secure architectures using software patterns*", Wiley Series on Software Design Patterns, 2013.

[Fla99] R.Flanders and E.B.Fernandez. "Data filter architecture pattern". In: *Proceedings of Pattern Languages of Programs Conference (PLoP) 1999*; 1999

[For04] B.A.Forouzan, "*Data Communication and Networking*" [Book] 2004. McGraw Hill. www.mhhe.com

[Gam94] E.Gamma, R.Helm, R.Johnson, and J.Vlissides, "*Design Patterns: Elements of Reusable Object-Oriented Software*", Addison-Wesley Professional, 1994

[Gar09] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández and E. Vázquez "Anomaly-based network intrusion detection: Techniques, systems and Challenges", *Computers & Security* Volume 28, Issues 1-2, FebruaryMarch 2009, Pages 18–28

[Gra00] C.Grace, "Understanding Intrusion Detection Systems": IT Journalist PC Network Advisor – Tutorial, 2000.

http://www.techsupportalert.com/pdf/t1523.pdf. Last Accessed on December, 21, 2013

[Hal06] S.T.Halkidis, A.Chatzigeorgiou and G.Stephanides. "A qualitative analysis of software security patterns". *Computers & Security*, Volume 25, Issue 5, July 2006, Pages 379–392

[Has09] K.Hashizume and E.B.Fernandez. "Symmetric encryption and XML encryption Patterns". In: *Proceedings of the 16th conference on pattern languages of Programs (PLoP 2009)*; 2009.

[Hay00] V.Hays, M.Loutrel and E.B.Fernandez . “The object filter and access control framework”. In: *Proceedings of pattern languages of programs (PLoP2000) conference*; 2000.

[Hey07] K.Heyman, “A new vrtual private network for today’s mobile world”, *Computer*, IEEE, December 2007, 17-19.

[How02] M.Howard, and D.Leblanc. “*Writing Secure Code*” Microsoft Series, 2nd edition, 2003.

[HP] HP-UX IPSec.

https://h20392.www2.hp.com/portal/swdepot/displayProductInfo.do?productNumber=J4256AA. Last Accessed on December, 21, 2013

[IBM99] IBM Redbooks “AS/400 Internet Security: Implementing AS/400 Virtual Private Networks” Publish Date: December 14, 1999. http://www.redbooks.ibm.com/redbooks/pdfs/sg245404.pdf Last Accessed on December, 21, 2013

[Igu06] V.M. Igure, S.A.Laughter and R.D.Williams, “ Security Issues in SCADA networks” *Computers and Security*, Volume 25, Issue 7, October 2006, Pages 498-506.

[Kis07] P.C.Kishore Raja, Dr.M.Suganthi.and R.Sunder “Wireless node behavior based intrusion detection using genetic algorithm” *Ubiquitous Computing and Communication Journal*., Special Issue, November 2007.

[Kum10] A.Kumar and E.B.Fernandez, "Security patterns for Virtual Private Networks", *8th Latin American Conference on Pattern Languages of Programs (SugarLoafPLoP 2010)*, Salvador, Bahia, Brazil, Sept 23-26, 2010

[Kum12a] A.Kumar and E.B.Fernandez, “Security Patterns for Intrusion Detection Systems”, *First International Symposium on Software Architecture and*

Patterns, in conjunction with the 10th Latin American and Carribean conference for Engineering and Technology, July 23-27, 2012. Panama City, Panama.

http://www.laccei.org/LACCEI2012-Panama/TechnicalPapers/TP010.pdf

[Kum12b] A.Kumar and E.B.Fernandez, "Security Pattern for the Transport Layer Security (TLS) protocol".*Proceedings of 19th International Conference on Pattern Languages of Programs (PLoP)*, 2012.

[Leh01] S.Lehtonen and J.Pärssinen. "A pattern language for key management". In: *Proceedings of pattern languages of programs (PLoP2001)* Conference: 2001.

[Lev04] N.Leveson "A new accident model for engineering safer systems", *Safety Science* 42(4): P237-270.

[Lid] Linux Intrusion Detection System: http://www.lids.org/ Last Accessed on December, 21, 2013

[Lu08] M.Lu and M.Weiss. "Patterns for Grid Security". *Mini-PLoP at Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA)*, 2008.

[Moz] Mozilla Newsgroup: mozilla.dev.tech.crypto

http://www.mozilla.org/projects/security/pki/nss/ssl/. Last Accessed on December, 21, 2013

[MS] Microsoft Forums: What is TLS/SSL

http://technet.microsoft.com/en-us/library/cc784450(v=WS.10).aspx. Last Accessed on December, 21, 2013

[Nok01] Nokia Inc., White Paper: "Combining Network Intrusion Detection with Firewalls for Maximum Perimeter Protection", April 2001

http://www.itu.dk/courses/DSK/F2003/Combining_IDS_with_Firewall.pdf. Last Accessed on December, 21, 2013

[Ope] Opera:Core concerns

http://my.opera.com/core/blog/2009/02/25/new-in-opera-presto-2-2-tls-1-2-support. Last Accessed on December, 21, 2013

[Pel07] J. Pelaez, E.B.Fernandez and M.M.Larrondo-Petrie, "Misuse patterns in VoIP", *Security and Communication Networks Journal*. Wiley, vol. 2, No 2, 635-653, published online**:** 15 Apr 2009. http://onlinelibrary.wiley.com/doi/10.1002/sec.105/pdf

[Rs] IBM: Real Secure Intrusion Detection Systems by IBM.

http://www.ibm.com/ru/services/iss/pdf/realsecure-server-sensor-ss.pdf. Last Accessed on December, 21, 2013

[Sch06] M.Schumacher, E.B.Fernandez, D.Hybertson, F.Buschmann, and P. Sommerlad, "*Security Patterns: Integrating security and systems engineering*", Wiley 2006

[Sel12] L.Seltzer, "Best practices and applications of TLS/SSL", white paper, Symantec Corp., 2012. http://resources.idgenterprise.com/original/AST-0036092_Best_Practices_and_Applications_of_TLS_SSL.pdf. Last Accessed on December, 21, 2013

[Sonic] Sonic Wall VPN products:

http://www.sonicwall.com/us/products/472.html. Last Accessed on December, 21, 2013

[Sta03] W.Stallings, "*Cryptography and network security: Principles and practice*" (3rd Edition), Prentice-Hall, 2003

[Sta12] W.Stallings and L.Brown, "*Computer security: Principles and practice*" (2nd Ed.), Pearson 2012.

[Sto06] K.Stouffer, J.Falco, and K.Kent, "Guide to supervisory control and data acquisition (SCADA) and industrial control systems security*", Spec. Pub. 800-82, National Institute of Standards and Technology (NIST)*, http://csrc.nist.gov/publications/drafts/800-82/Draft-SP800-82.pdf. Last Accessed on December, 21, 2013

[Uzu12] A.V.Uzunov, E.B.Fernandez and K.Falkner "Securing distributed systems using patterns: A survey" *Computers & Security*, Volume 31, Issue 5, July 2012, Pages 681–703

[Uzu13] A.V.Uzunov and E.B.Fernandez, "Cryptography-based Security Patterns and Security Solution Frames for Networked and Distributed Systems", Submitted for Publication

[Ver02] T.Verwoerd and R.Hunt. "Intrusion detection techniques and approaches", *Computer Communications*, Volume 25, Issue 15, 15 September 2002, Pages 1356-1365.

[Van09] M.VanHilst, E.B.Fernandez and F.Braz, "A Multi-dimensional Classification for Users of Security Patterns", *Journal of Research and Practice in Information Technology*, Volume 41, Issue 2, 2009.

[Wei02] Q.Wei and S.Srinivas, "IPSec-based secure wireless virtual private Network". *Milcom 2002 Proceedings* Volume: 2, Pages 1107- 1112.

[WikIP] Wikipedia IPSec http://en.wikipedia.org/wiki/IPsec, Last Accessed on December, 21, 2013

[WikTLS] Proposed Standard: The Transport Layer Security (TLS) Protocol Version 1.2 http://wiki.tools.ietf.org/html/rfc5246 Last Accessed on December, 21, 2013

[Yas04] A.Yasinsac and J.Childs. "Formal analysis of modern security protocols", *Information Sciences* Volume 171, Issues 1–3, 4 March 2005

[Yod97] J.Yoder and J.Barcalow. “Architectural patterns for enabling application security”. In: *Proceedings of fourth conference on Pattern Languages of Programs (PLoP ’97)*, vol. 51. Monticello, Illinois; 1997.