Web Application Security for Omni Channel Banking (OCB)

Submitted By

Binaka Patel

16MCEI14

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2018

Web Application Security for Omni Channel Banking (OCB)

Major Project

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering (Information and Network Security)

Submitted By

Binaka Patel

(16MCEI14)

Guided By

Dr. Vijay Ukani

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2018

Certificate

This is to certify that the major project entitled **"Web Application Security for Omni Channel Banking (OCB)"** submitted by **Binaka Patel (16MCEI14)**, towards the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering (Information and Network Security) of Nirma University, Ahmedabad, is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project part-I and part-II, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Dr. Vijay Ukani
Guide & Associate Professor,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr. Sharada Valiveti
Associate Professor,
Coordinator M.Tech - CSE (INS)
Institute of Technology,
Nirma University, Ahmedabad

Dr. Sanjay Garg
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr Alka Mahajan
Director,
Institute of Technology,
Nirma University, Ahmedabad

Statement of Originality

I, **Binaka Patel**, **16MCEI14**, give undertaking that the Major Project entitled "**Web Application Security for Omni Channel Banking (OCB)**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science & Engineering (Information and Network Security)** of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

_____

Signature of Student

Date:

Place:

Endorsed by

Dr. Vijay Ukani

(Signature of Guide)

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Dr. Vijay Ukani**, Associate Professor, Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout this work. The appreciation and continual support he has imparted has been a great motivation to me in reaching a higher goal. His guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

It gives me an immense pleasure to thank **Dr. Sanjay Garg**, Hon'ble Head of Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to **Dr. Alka Mahajan**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout course of this work.

I would also thank the Institution, all faculty members of Computer Science and Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

- Binaka Patel

16MCEI14

Abstract

Today an internet has so many different web applications. One class of things that can cause a great deal of issues are security holes, because of programming mistakes, inexperience and weak system protection. Attacks against the financial business are winding up progressively advanced and very focused on. This thesis focus on threat analysis of the omni channel banking system, A brief introduction about threat modelling and implementing two methods STRIDE and Attack tree into system. Combining the both methods is proposed here. It's enhance the effectiveness of the threat analysis incredibly and furthermore has great practicability. Applying combination of the two methods into omni channel banking gives the detailed threat analysis and try to cover the new threats which are not covered into single analysis methods. So, The Omni channel banking give less vulnerability to threat than other online banking system.

Abbreviations

OCB	Omni Channel Banking
STRIDE	Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of privilege
CAPEC	Common Attack Pattern Enumeration and Classication
DFD	Data Flow Diagram
OData	Operational Data Protocol
CSRF	Cross-Site Request Forgery
API	Application programming interface
JPA	Java Persistence API
ISO	International Organization for Standardization
IEC	International Electrotechnical Commission
OASIS	Organization for the Advancement of Structured Information Standards

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Web Application Security

A web browser is a medium for any web application to run on any platform which require network such as intranet or internet. Web browser act as a client for web applications. Due to web application, thousands of clients do not have to install any software on their machine. Therefore, clients have ability to maintain and get timely update on web applications. There are many types of web application in the market including Business application, Online Banking, E-commerce application, Web mail etc. There is aim for any web application to provide some critical and useful services to the customer. There are some issues with the web application but main issue for any web application is related to security.

Internet is called as open source system. Most of all attacks are related internet therefore security of web applications is mandatory and it creates threat to many users [1]. Some web application is more interactive and it requires to exchange the information which is sensitive information such as credit card, debit card, health, financial numbers etc. Database can store all the information of any application. If web application is not secured, then database is also not secured.

Web application on the internet is increasing rapidly. From that, many application does not contain security or it contain less security which is not enough for web application to survive from attacker. Figure 1.1 shows the Recent Scenario of web Application security today. Web application security is concern for the owners and the host system.

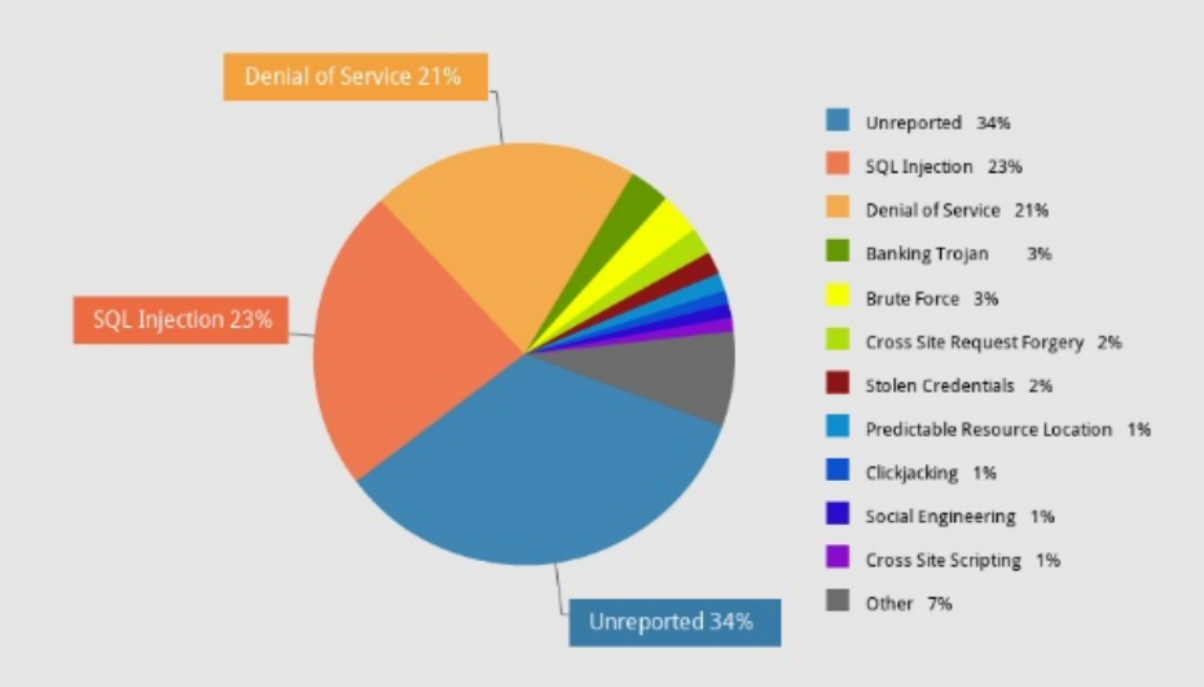

Figure 1.1: Recent Scenario of Web Application Security[2]

Attacker can attack on web application in different way such as installing anything on machine, automatic downloading, spamming etc. While making web application, owner may not think about why attacks are done, Owner may think that any attacks will not be successful.

1.2 Omni Channel Banking

Main motive for Omni-channel banking is to provide coherent view of the bank to the customer at any time or any place. In the core banking, customers are handled by online banking, ATM machine and its branches. From the survey, nearly half of the customer of Omni-channel banking have used all three channels in past sixth months [3]. Increasing usage of online and mobile banking rapidly, customer of Omni-channel banking will likely to expand it simultaneously. In the 2012, there are 31% customers for Omni-channel banking but in the 2014, there are 41% customers for Omni-channel banking which showing us increase in accessibility of the web application [3]. Any medium for interaction is selected by the customer will provide by the Omni-channel banking which allows customer optimal channel experience and consistent.

Omni channel banking gives us banking services which are easily accessible. With the help of much advanced and whose interface is user friendly, the application which

is particularly used standalone or a thing used for net surfing, to connect to the banks system, people can use the Internet. This expanding trend wants to say the same that the issues related to security for integrity, confidentiality and secrecy are becoming more successful and serious in case of online banking systems for both, to the banks and customers. Risk evaluation and threat mining study of omni channel banking are receiving extensive attention [4].

Developers need to consider all possible threats for Omni-channel banking after only, they can provide sufficient security to handle threats. otherwise they cant provide good level of security. Lack of security leads a system to vulnerable to access the information easily. For analysis and security requirements, modeling of threats play an important role. The threats modelling involves all aspects of security issues and most of the possible attacks. After determining the security requirements must be present in the system, develops can follows the SDLC life cycle for development.

1.3 Threat Modeling

Threat modeling is one of the way which provides the security while developing any application [5]. This model should be applied at the starting of software development as it will help developers in different ways. It verifies the architecture of application, various threats related to the application, steps to minimize or prevent the attacks will be carried out with the help of testing with respect to threat model.

Threat modelling becomes the important module while developing any kind of software. However, this process often gets abandon by not involving much resources so that they can deeply study and recognize the threats in the system. But in the overall process, threat modelling helps to safeguard the sensitive information included in online banking system.

Threat modelling states secured methods for securing application. There's one method is to prevent CSRF attack. For preventing CSRF attack other techniques are used, here we proposed the technique to preventing CSRF attack through OData protocol.

1.4 OData Through Preventing CSRF Attack

Operational Data Protocol (OData) is ISO/IEC approved, OASIS is a standard which defines collection of best trials for building and utilizing RESTful APIs [6]. OData

is very helpful to focus on business-related logic in process of RESTful APIs without having tension of various ways to define request and response headers, status codes, URL conventions, media types, HTTP methods, query options, payload formats, etc. In addition to this OData provides a proper approach for pointing out the actions/functions, tracking changes for re-using the methods and sending the batch/asynchronous requests. Easy to consume OData is RESTful APIs. The metadata of OData, machine can read the description of the data models of APIs, starts the creation of strong generic client proxies and tools. In this case OData acquires Java Persistence API (JPA) [7]. OData URLs contain sensitive data which is secured through HTTPS.

CSRF (Cross site request forgery) attack is an occurs when a malicious or program cause user's web browser for performing unwanted actions on a certified site for which user is authenticated. To prevent this attack we used CSRF TokenKey concept into OCB. Application generates session key at user's logging time and stored for user's session [8]. when user's session will expired then session key expired. If any changes detected into CSRF TokenKey at request than session will expired automatically. Here CSRF TokenKey generated using OData into application.

1.5 Summary

Web Application Security has attract to significant attention as humans increasingly dependent on Internet. So here need for designing security into web application rather than provide it as an afterthought has been discussed. Omni channel banking application is no less secure than traditional online banking system. For that different threat analysis methods given different features that can be used by threat modeller. Also diffenrent methods stated in threat model, Application will implement that techniques into OCB. Here we introduced OData and how to use of that we can prevent CSRF attack.

Chapter 2

Literature Survey

Customers information is the most important element to be secured in omni channel banking system. The users are the most delicate link in the whole OCB to be attacked. As the whole system of omni channel banking was provide with the highest security as even some small attacks could bring the whole system down which could cost unexpected. In omni channel banking system the banks server is full equipped with the highest security level techniques, so the OCB server has less prone to attacks [9].

In a related study, the Naval Research Laboratory researched approximately 50 security flaws and analyzed that 50% flaws of security were caused by the specification or requirement flaws [10]. The charge for designing security into software applications rather than retrotting it as a review has been additional discussion. While the accent of starting threat modeling during the requirements analysis appearance has been additional discussed in the literatures existing clay notations such as data-ow diagram and graph trees, are abundantly focused on the architecture and development phases.

Threat modelling with the four-step framework hypothesizes by Adam Shostack [5]. This framework should assistance best threat model to accomplish compact threat models of their system. He has briey stated some history of threat model as accomplished at Microsoft. It shares size of the accepted action and some inform learned.

Ebenezer A. Oladimeji presents an goal-oriented methodology for demonstrating what's more separating security threats throughout engineering of requirements. A greater amount specically, those recommended support [11] controlling those threat demonstrating also separation methodology utilizing those notions of negative soft goals to observe security threats and inverse assurances for relief analysis [11], the utilization of graphical

model components for documenting the whole risk of demonstrating process, and the observation of bases to select around alternative countermeasures.

Mohamed Abomhara [12] et al. objectified an risk model change for telehealth frameworks utilizing Microsoft risk representative device around 2014 might have been built. And meanwhile they get ready for risk moderation, framework assets, threat agents, unfriendly actions, threat furthermore their impacts and also poor for countermeasures were analyzed and identied.

Kenneth Edge [13] et al. defined advance support risk and discussed how they can be implemented in the risk analyze of an online cyberbanking system. Also they authentic and equivalences to bear the metrics up the risks were developed.

Suvda Myagmar defined advance protection attacks and discussed how they can be implemented in the specification of an online cyberbanking system [14]. Additionally they authentic and equations to investigate how threat model can be adapted as foundations for the specification requirements. She analyzed the differences amid clay software articles and circuitous systems, additionally present three case studies of threat modeling: Software-Dened Radio, a network trafc modelling tool (VisFlowConnect), and a cluster security monitoring tool (NVisionCC).

STRIDE is very useful method of approaches for searching threats against computer systems. Using of STRIDE, we should make it easy to categorize and finding threats that normally would not see [12]. STRIDE as a threat modeling tool has seen as high level and it should be replaced by something more detailed [1]. STRIDE and Attack library are theoretical approaches for discovering threats. So, Henrik Andre Stene used Attack Tree method on their system. Attack Tree giving threat information in Graphical way while Other two methods CAPEC (Attack Library) and STRIDE doesnt. From Graphical view, its easy to understand about threats on system [3].

Normally for creating attack tree we used Attack Library. May be Attack tree is a great tool for presenting threat, it doesnt view itself as the most efficient method when trying to finding new possible threats [14]. Attack trees is appear to be a tool which is more suited during the threat mitigation process.

2.1 Motivation

The motive of putting the bank system online or Omni channel banking was to improve the performance and enlarge the industrial management as well as the operations carried out. It mostly gives easy access to various services been performed in system which turns out to be user friendly. Internet is one of the best way to connect to the bank where any kind of transactions are possible within a second. Most importantly it saves a lot of time of users but the question arises of security. As more and more users are using online banking, Omni channel banking (OCB) need to give certain security aspects so that users could manage to operate with OCB instead of normal banking [15]. The layered architecture should be implemented to maintain the security and privacy of everyone. It is the main objective to focus on the security element. So, in the given thesis, Threat Analysis is the highlight and identifying the threats related to software and hardware using of threat modeling.

Threat analysis has many methods to finding the system's threat, but they are more time consuming to developing and also a more complex to understanding a developer. STRIDE is very useful method of approaches for searching threats against computer systems. But it has seen as high level and it should be replaced by something more detailed. STRIDE and Attack library are theoretical approaches for discovering threats. So, Henrik Andre Stene [9] used Attack Tree method on their system. Because its giving threat information in graphical way. Normally for creating attack tree we used Attack Library. If Attack library is not updated with new attack than our attack tree gone wrong. So we proposed combination of STRIDE and Attack tree, which giving more security to system and try to cover all possible threat.

2.2 Goal

The Goal behind this thesis is to create new threat analysis method using of STRIDE Threat model and Attack tree method. then we apply that method into Omni channel banking system and analysis the threats. Also implement one technique into OCB for preventing CSRF attack. For that we work on following manner:

- The process of threat analysis is provided briefly for characterizing threat information which is coming from various threat methods, for the better understanding of

all types of threats to a omni channel banking.

- The threat to the omni channel banking was analyzed with the help of Microsoft Threat Modeling Tool 2014 [13], and Constructing the Data Flow Diagrams of the system.

- Using of that DFD diagram representing a system threat analysis method with the STRIDE threat model and Attack Tree for threat analysis.

- Combining both STRIDE and Attack Tree models, Creating new model for system which is less complex and easy to understand at designing phase. For that we First list out the threats using of STRIDE than creating Attack tree by layer by layer decomposition.

- Implementing OData into application. Then generating CSRF TokenKey for user's secured session into application using of OData.

2.3 Summary

An experienced threat modeller can be utilize STRIDE threat model, Attack Trees, and Attack Library. But Lerner will most likely not find many threats using Attack Library. Because if Attack library is not updated with new threats then using of it, created attack tree are missing that attack. So here proposed is to constructing new method which giving combination of STRIDE and Attack Tree. After Applying this threat analysis method to Omni Channel banking system threat analysis, we design the more effective threat model for OCB. Also we gives the security to OCB from CSRF attack using of X-CSRF TokenKey through OData.

Chapter 3

Threat Modelling

Threat modelling is theoretical way for finding security holes and security error in program. After applying it correctly on system it will help in design and implement the more secured program and system [13]. In every program or system must implement threat model because it gives surety they cover the specified security aspects.

Using of threat modelling into system, Developers understand how to identifying different threats and doing secure coding so the different threats could be avoided by system.

3.1 Threat, Risk and Mitigation

Threat is a harmful and unwanted incident of system [5]. An attack is considering as a threat here because of due to attack on system its result harm on system. For example to Customer claiming to have not performed Transaction but transaction made from their account, its harm on Repudiation of system.

Risk is characterized as the impact of vulnerability on targets and the level of risk is measured as a segment between the outcome of a risk being executed and the probability of the danger being executed [12].

The meaning of word Mitigation is the halting the likelihood of a risk being executed. To preventing the threats we perform some actions, its called mitigation in simple language [12].

3.2 The Four-Step Framework

From the recent study Adam Shostack [5] have concluded that threat modellers should look at threat modelling as four steps in Figure 3.1 that each contain sub goals and that by completing all four steps we can make sure that a system is satisfactorily safe.

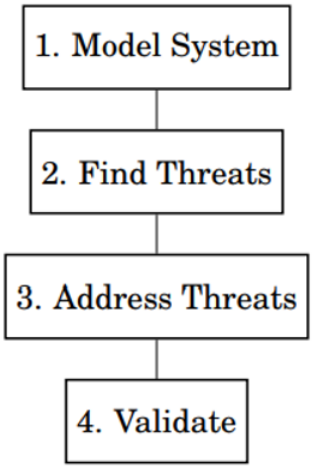

Figure 3.1: Four-Step Framework

This framework comprise of four question that the threat modelers and engineers need to consider [5], and these question can be found in bellow. By noting every one of these question, well ordered, the threat modeling group ought to have the capacity to make a strong threat model of their system.

- What are you building?
- What can go wrong with it once its built?
- What should you do about those things that can go wrong?
- Did you do a decent job of analysis?

3.2.1 Model System

All product venture comprises of various sorts of detail and Documents. Some will be intricately demonstrated, some will generally comprise of a composed depiction of system necessities and client wishes. In the rst step in the four stage structure, the threat modeler

should recognize what they are building. One of the simple methods for getting a review is by drawing data ow diagram, or other visual models of the system. By taking a gander at such diagram, the threat demonstrating team ought to have the capacity to get outline of how broad the system truly is.

3.2.2 Find Threats

Stage two in the four step system is the production of the threat model. By taking a gander at different diagrams of the system, the threat modeler or threat modelling team will have the capacity to discover at least one or more assault designs that might be threat against system. For this progression, it is proposed to use one or more threat modelling techniques to nd dangers.

3.2.3 Address Threats

In the wake of finding each conceivable threat they could consider, it is the ideal opportunity for stage three. Stage three comprises of what to do with each danger, and how to conceivably palliate the distinctive threat. Many Product Development ventures having constrained resources like time or cash, and it is in this progression that the threat modellers need to remain on which assaults to mitigate, and which assaults are so dark, difficult to execute, or just not that harming to the system if executed, that they can be disregarded. This Phase we have to address threats.

3.2.4 Validate

After every single conceivable threats have been evacuated or been considered as not dangerous, the time has come to re-assess the system design and implementation. threat displaying is thought to be a redundant procedure, and if the approval of the system fail, at that point the threat modelling process needs to return back to step one or step two.

3.3 Methods For Threat Modeling

Many different Techniques are used to finding threats. All have their advantages and disadvantages, and different techniques work in different circumstance. There are many approaches to finding threats. Each has own advantages and disadvantages, and different approaches may work in different circumstances. In this chapter, we will learn about the following Methods to finding threats:

- STRIDE
- Attack Libraries
- Attack Tress

3.4 STRIDE

STRIDE introduce by Loren Kohnfelder and Praerit Garg in 1999[1]. The STRIDE stands for Spoofing, Tampering of data, Repudiation, Information Disclosure, Denial of Service and Elevation of privilege. The greater part of this words is for an entire gathering of threats. It is less easier to find the threat against certain part of a PC by this sort of classification. To comprehend STRIDE as an approach for finding threats first we have to perceive what each letter in the memory aide is a case of and what it should speak to in our product. As a matter of first importance, all aspects of the STRIDE mnemonic is a quality that we don't need our program or framework to have. Using STRIDE we can easily categorize and discover threats, which are normally hard to find. Table 3.1 categorize the threats based on property and giving the definition of threats which are affected to this property.

Threat	Property	Definition
Spoofing	Authentication	imitation something or another person
Tampering	Integrity	Modifying the data or source code
Repudiation	Non-Repudiation	Claiming to have not performed activity
Information Disclosure	Confidentiality	Presenting data to somebody who not authorized to see it
Denial Of Service	Availability	Deny or corrupt the system to clients
Elevation Of Privilege	Authorization	Access capabilities without Authorization

Table 3.1: STRIDE Threats [5]

Now we have to know what kind of property have to handle in our model, For example, bank database having all necessary details about customers. That data are confidential. Following are some threats against a database is:

- Someone puts on a show to be a customer service agent, to access the database.
- A disappointed worker with malicious expectation chooses to change all the telephone numbers to the bank clients in the database.
- The same disappointed worker denies having changed telephone numbers to all clients.
- Some undisclosed files containing the bank future worker termination plan is made accessible to all workers.
- The database is made inaccessible because of high load.
- Customers are offered rights to peruse records not implied for them.

3.4.1 Violation of Authentication

We may know that breach in authentication is a someone using another persons user name and password to access data that he or she not supposed to access, but authentic problems can be much more than this. Spoong implies somebody guaranteeing identity of another. This can get to records of your coworker utilizing his secret password, to asserting you are the legal counselor of an as of late expired individual from India honorable with no beneficiary. In This kind of cases somebody hide their actual personality, while performing, or claiming to perform, errands that they are not authenticated to do.

3.4.2 Violation of Integrity

The real question in information security is to know that file is real or it has been modified or just to placed here for a trap. Tampering with data is a big problem for all kinds of systems. If systems dissatisfied employee change all phone numbers registered to all the customers in the client database, then the client database is no more for uses because of it containing false data. Tampering is modifying the data which is on disk, also attacker can modifying the data which are on network or in memory.

3.4.3 Violation of Non-Repudiation

"I didn't do it" is a typical sentence to hear when something has turned out badly. Non repudiation, or responsibility, is where it isn't conceivable to debate a demonstration, such as claiming it was not you who messed with all the telephone numbers in the client

database. This is a urgent piece of data security, to have the capacity to guarantee responsibility in a framework. For all clients, heads or other individuals with get to, it ought to be workable for them to demonstrate they did or did not accomplish something. The most straightforward approach to guarantee that it is conceivable to have non-repudiation in a framework is to have, hold and investigate logs for each activity. Likewise with the disappointed coworker who denies having changed all the telephone numbers in the database, by logging who did really submit the adjustments in the database logs, the nancial organization ought to have the capacity to invalidate the announcement made by worker with steady confirmation.

3.4.4 Violation of Confidentiality

Violation of confidentially perceivedas disclosing of something by person A to person B, that person B should not know or does not have authorization to know, but its possible to not on purpose to uncover information to someone without the true authorizations. Despite the fact that we have to log things occurring in our system we ought to be deliberately in choosing what to log and what information we can skip from each log section. Information Disclosure can be occur by accidently composing secreat information to log records, which having lower approval level than the first le. In the event that the manager in any organization tries to spare a le to his backup server named End Letter for Mark.docx yet it comes up short and a log section is put in the logs saying ERROR: Could not spare "End Letter for Mark.docx", the plate is full, at that point there is a data spill. The le name itself is information that should be delicate and just open from the Admin PC or client account, yet it is presently available to any individual who has permission to see the backup plates error logs.

3.4.5 Violation of Availability

Good services are main reason for any business to make money. If the client database is not available, then clients will unable to purchase anything from the website. To banking this might be services such as loan applications, selling insurance policies and opening accounts. In any business, Denial of service is a very serious threat and its performed intentionally or unintentionally. It is therefore important that we secured our system against these type of threat. Other threat which perform violaton of availability is Distributed Denial of Service (DDOS) attacks.

3.4.6 Violation of Authorization

It is important to discover and mitigate this type of threat. Elevation of Privileges are give permission to do perform any action which they are unauthorized to do. For example, someone running code on a system as a administrator. Two types of authorization related violations are there. May be its gain by corrupting a process into system. But its perform successfully if user having some control of system. The second kind of violation of authorization is on the grounds that the system has carriage get to control checks or it might not have any entrance control checks all together.

3.4.7 Data Flow Diagrams into STRIDE

STRIDE is a threat modeling tool which is used Data Flow Diagrams (DFDs). DFDs is system diagram that make easy to understand that which part of computer program that communicate with and direction of communication [11]. The data is passed on to only part of the system which have higher privilege, Threat to data is less.

In threat modeling, the Data Flow Diagram (DFD) is generally used to mirror the information stream connection between omni channel banking an account framework and outside interactors. Figure 3.2 shows example of Data Flow Diagram and its symbols which are using in analysis of threats in threat modeling.

The main principle is to disintegrate the system or the computer program into parts and check for every relevant threat. Usually DFD contains data flows, Process, External entity and data-store. But in threat modelling for system we added other element which is called trust boundary. Trust boundaries are represented by dotted line covering some parts or part of system. Trust boundaries is boundaries between trusted and untrusted parts of the system. Untrusted parts denotes the part with lower privilege or might be that part is completely public. For example, Banking clients have to give proof before logging into system, they are not directly gain access to account they authorize themselves first. Here trust boundaries makes easier to differentiate the parts of system.

3.5 Attack Libraries

Detailed lists of threats are called Attack Libraries or reference tables [5]. If we would have a list of threats or attacks, it would be more helpful to make model more complete. A fully detailed list is, in this case, a list containing all the possible threat to all the possible

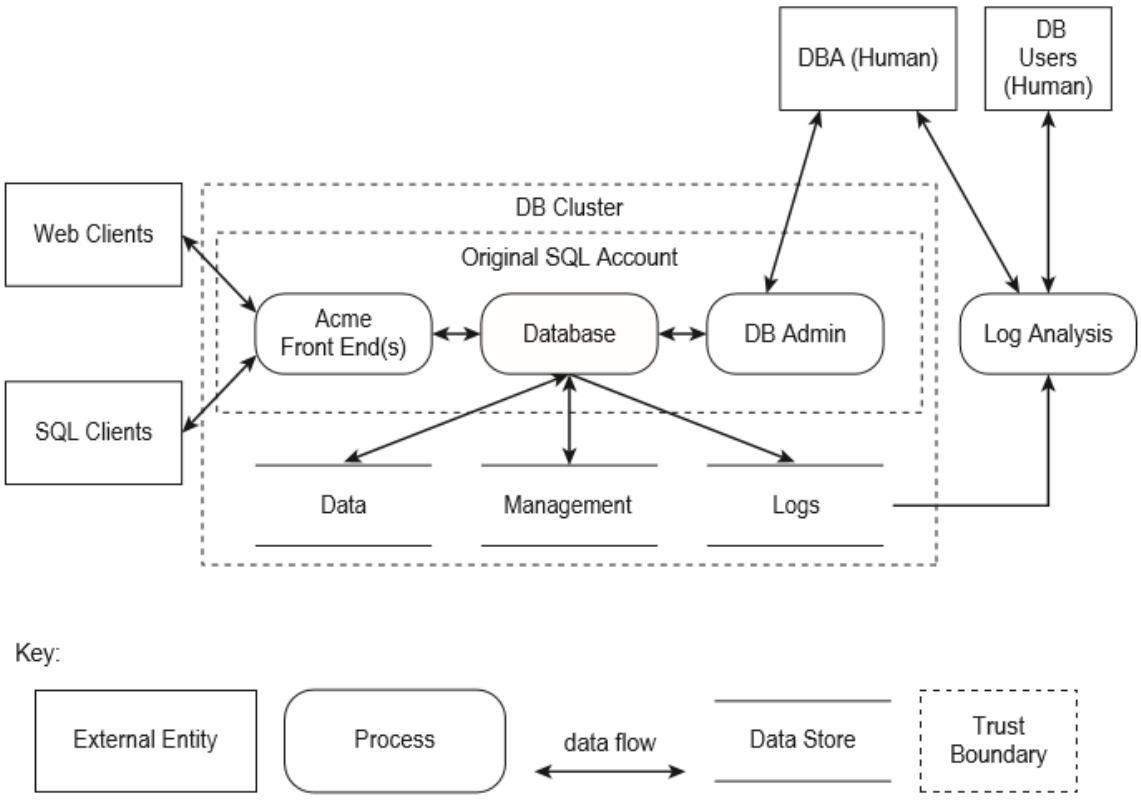

Figure 3.2: Basic DFD Diagram [1]

information in present and future. Pragmatically that list cannot be an unlimited number of every possible threats, however there can be libraries which are closing on a practical limit. Consideration of Scope of list and its audience is very important at the time of creating attack library. If the analyst is security expert than dont explain all small information, also when our scope is to communicate the threats that time dont create a list which have hardware security threats. There is One limit of attack libraries is that they are more time consuming to develop. Developing a new library demand more time investment, That is the main reason of lesser number of library. So There is available one community created list called Common Attack Pattern Enumeration and Classication (CAPEC) [15]. This is a large, detailed and highly structured list yet more efficient.

MITRE Organization working on different aspects of security, They created Common Attack Pattern Enumeration and Classication CAPEC is an attack library. In this list 463 different attack patterns are there and they organized into 16 attack categories. Adam Shostack while commenting about CAPEC said that If reviewer take an aerage of 5 minutes for each 475 entries, that would be still 40 hours of work. List of 16 Attack categories in CAPEC are following [10]:

- Gather Information
- Deplete Resources
- Injection
- Deceptive Interactions
- Manipulate Timing and State
- Abuse of Functionality
- Probabilistic Techniques
- Exploitation of Authentication
- Exploitation of Authorization
- Manipulate Data Structures
- Manipulate Resources
- Analyze Target
- Gain Physical Access
- Malicious Code Execution
- Alter System Components
- Manipulate System Users

Attack libraries is not providing blue print for threat model but it gives whole perspective to threat modelling compared to STRIDE. Using abstract approaches like STRIDE, we have to think about of evry possible threat ourselves and this might be very time consuming but with the help of detailed libraries it would be easier to find threats by listing them.

3.6 Attack Trees

STRIDE and Attack Library are the theoretical methods to finding the threats, But Attack tree is represent the graphical path to finding threats [5]. Also we can create the general attack tree patterns which are applied to different systems. Common way of attack tree is using existing attack tree on system or creating particular trees for system. For creating of attack tree we consider DFD of system or Attack Library. At that point we emphasize over every node in the threat tree and check whether that node is a danger against our system and after that repeat this for all nodes of the system and every one of the trees important to our system.

Bruce Schneier wrote an article about attack trees in 1999 where he clarifies them in a simple way, "threat trees give a formal, orderly method for depicting the security of system, in view of changing assaults. Fundamentally, you speak to assaults against a system in a tree structure, with the objective as the root node and distinctive methods for accomplishing that objective as leaf nodes [9]." For every risk, or objective as Schneier puts it, we make a root node in another tree. This root node speak to the danger we are endeavoring to execute and for every child in the tree it speaks to a sub risk or a method for executing the risk. In a attack tree we have two sorts of nodes. We have AND-node and OR nodes. At the point when a node is an OR node we have the likelihood to pick both of them to accomplish our objective or subgoal. However when in AND-nodes we need to take every one of them together for accomplish our objective or subgoal. In the event that Detection of security flows wind up noticeably simpler by utilizing attack tree. The issue with attack trees is however that making new trees might be a repetitive and tedious errand.

3.6.1 Creating the Attack Trees

While creating new attack trees we need to consider what kind of tree it is supposed to be. We can create AND-trees or OR-trees. The type of tree decides its representation. In most cases we will create OR-trees. Figure 3.3 is an OR tree because the root node is an OR-node with two children that both is a possible way to achieve the goal and that they do not rely on each other [5]. The theoretical tree that we can see in Figure 3.4 is an AND-tree because both children to the root node has to be completed in order to achieve the goal [5]. Most attack trees will be OR trees since it often is possible to

execute a threat in many different and independent ways. After we have decided on a representation we need to choose a root node. The root node will contain the threat that we want to model. We continue to add subnodes for each way that we can execute the threat. One of the harder things with creating attack trees is the same problem that we have with attack libraries. When creating an attack library we need to keep it short enough so that it will be practical to use, but long enough so that we cover every probable threat. It is very important to keep the attack tree within a practical length, while also covering the threat. This problem is called completeness and there is no blueprint on when an attack tree is complete.

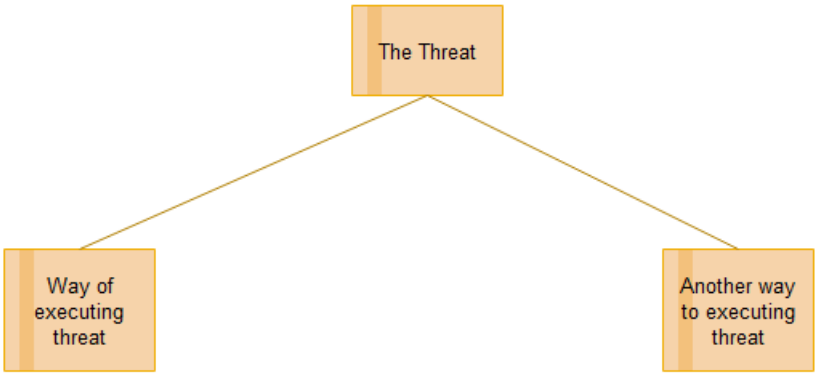

Figure 3.3: Attack Tree Using OR-node

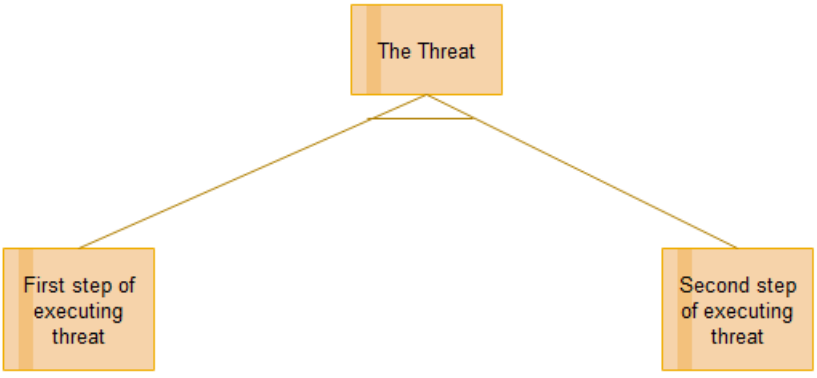

Figure 3.4: Attack Tree Using AND-node

While making new attack trees we have to consider what sort of tree it should be. We can make AND-trees, OR-trees. The kind of tree chooses its portrayal. Much of the time

we will make OR-trees. Figure 3.3 is an OR-tree in light of the fact that the root node is an OR-node with two child node, that both is a conceivable approach to accomplish the objective and that they don't depend on each other. The hypothetical tree that we can find in Figure 3.4 is an AND-tree in light of the fact that the two child nodes to the root node must be finished with a specific end goal to accomplish the objective. Most attack trees will be OR trees since it frequently is conceivable to execute a risk in a wide range of and autonomous ways. After we have settled on a portrayal we have to pick a root node. The root node will contain the risk that we need to show. One of the harder things with making attack trees is a similar issue that we have with attack libraries. While making an attack library we have to keep it sufficiently short with the goal that it will be down to earth to utilize, yet sufficiently long so we cover each likely danger. It is imperative to keep the attack tree inside a viable length, while additionally covering the threats. This issue is called fulfillment and there is no outline on when an attack tree is finished.

3.7 Summary

Threat modelling is important part of software development. Using of it we sure about the system is safe to use for the end user, and system is not vulnerable to common attack. The four-step framework is good practice to divide the whole task of threat modelling into four steps. We given brief overview of Methods for threat modelling.

STRIDE should help the threat modeling team to start finding threats against a certain part of the system. STRIDE is a modification where the type of system decides which of the 6 attack categories should consider. This will limit the search for threats down and make the task simpler.

Attack libraries are lists or a set of common threats that one should consider. There is no blueprint of how an attack library should be, so the variety ranges from short lists to lists contain several hundreds of threats.

An attack tree gives graphical representation of threats into system. It consist of a root node which gives specific threat and child-nodes contains way of executing the parent node threat. Attack tree using two types of attack trees: And and OR tree.

Chapter 4

Analysis of the OCB Threat based on STRIDE Model

In this chapter we present the data flow diagram to describe the omni channel banking business process in detail. and then implement STRIDE model using of that Diagrams.

4.1 Data Flow Analysis of OCB system

The omni channel banking system is a system that provides online banking services to customers, it is also a system that exposed to the Internet environment. The external entity of omni channel banking system mainly include:

- Omni channel banking Client(1.0): They can request to the omni channel banking system, they can be making request to the omni channel banking system, through browser and contract, etc .
- B2B/B2C System(2.0): They can request to the omni channel banking system, including the third party electronic payment platform and agency payment platform.
- Management stuff(3.0): They can be started by administration operation demand to the omni channel keeping money framework, OCB framework to give the relating administration interface to administration faculty to bear on the administration to the OCB framework, i.e. log review.
- Core bank account system (4.0): Suppose the omni channel banking internal network environment and itself are credible.

The main business operation of omni channel banking can be divided into :

- Login(5.1)
- Query type of Transaction (5.2)
- Set type of transaction (5.3)
- Transfer type of transaction (5.4)
- Electronic payment (5.5)
- Cash changing transaction (5.6)
- System management (5.7)

First level data flow diagram of omni channel banking business process operations is given into Figure 4.1. External entity have to first successfully perform login process. Also other Business operations can be performed by external entity. Here Account(5.8), Log record(5.9), and B2B/B2C data(5.10) are Data store.

Figure 4.2 to Figure 4.7 gives the second level data flow diagram of Business operation Respectively.

Figure 4.2 shows the Login flow for OCB Client. After the Authenticate(5.1.1), Authorize(5.1.2), Login Processing(5.1.3) OCB client should login into Application. For that process it used Account and Log Record Data store.

Figure 4.3 uses the Authentication(5.3.1), Authorization(5.3.2), and set(5.3.3) process to set type of transaction. Here Data stores Account and Log records are used. First Client set parameter for authentication process, using of Log record and Account information user identified. Request go to Authorize process now. In this process system checks user privileges and set the identity.After set process it gives result to OCB Client.

Figure 4.4 shows the DFD for the Transfer type of Transaction. OCB client first process for authentication(5.4.1), than request goes to Authorize(5.4.2) process, after success of this process user able to perform Transfer(5.4.3) process and got response.

For electronic payment process B2B/B2C system have to authenticate and authorize(5.5.1) itself. As shown in Figure 4.5 OCB client have to authenticate and authorize(5.5.3) itself for further operation in electronic payment. After this process uses of

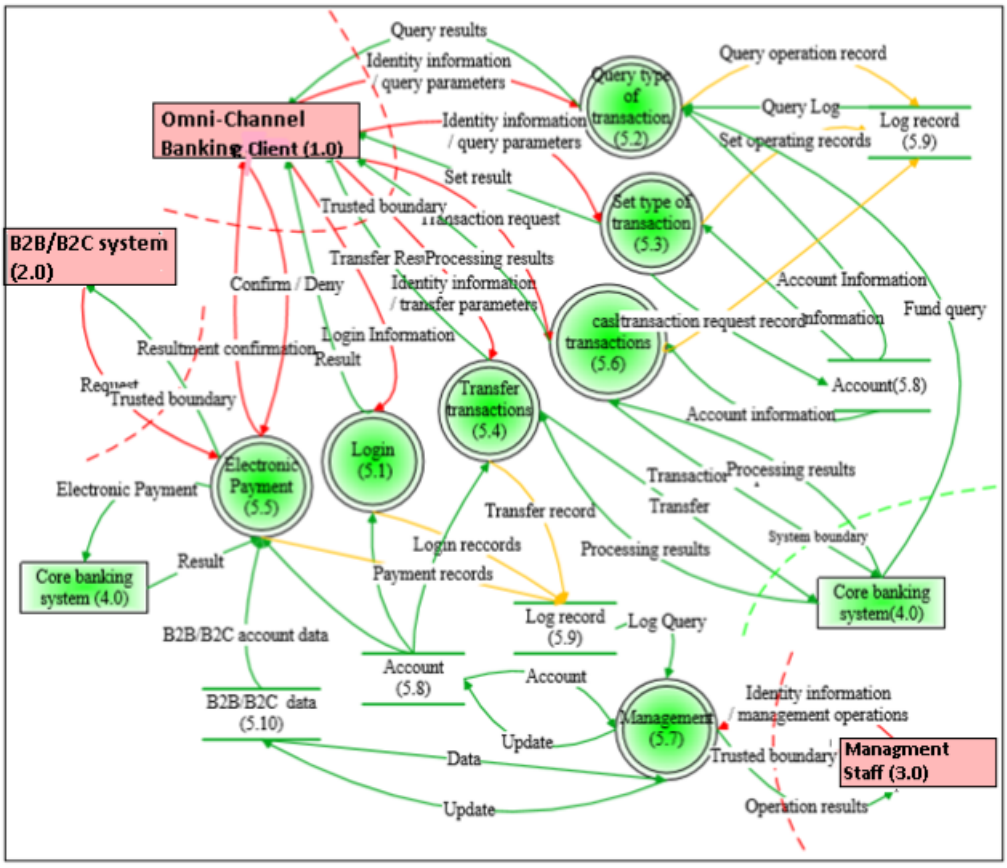

Figure 4.1: System DFD

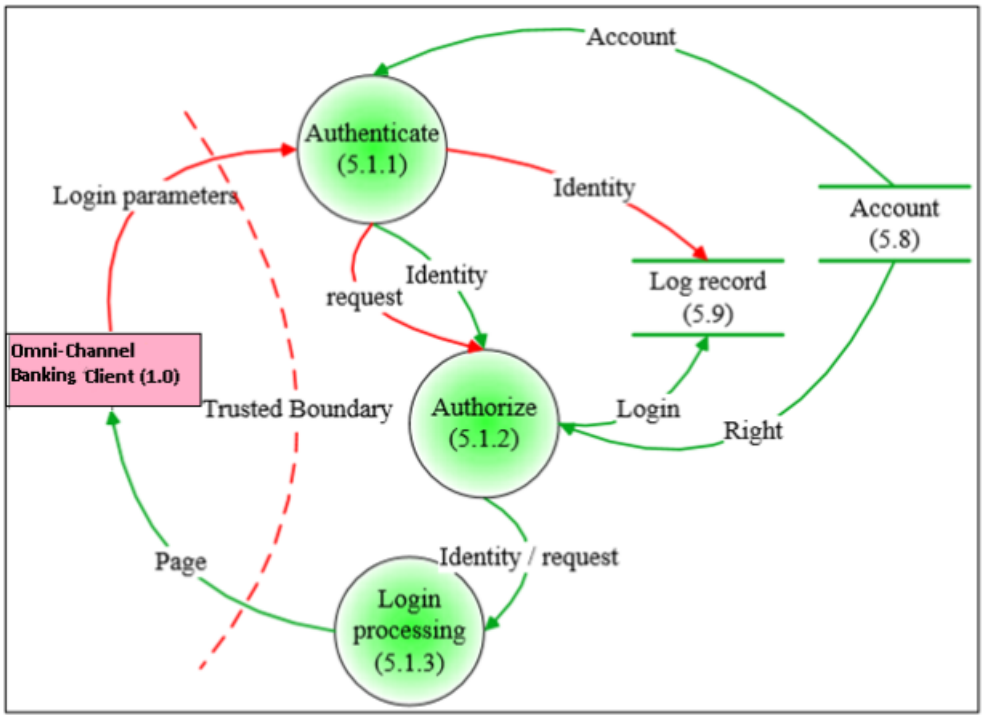

Figure 4.2: Login Flow

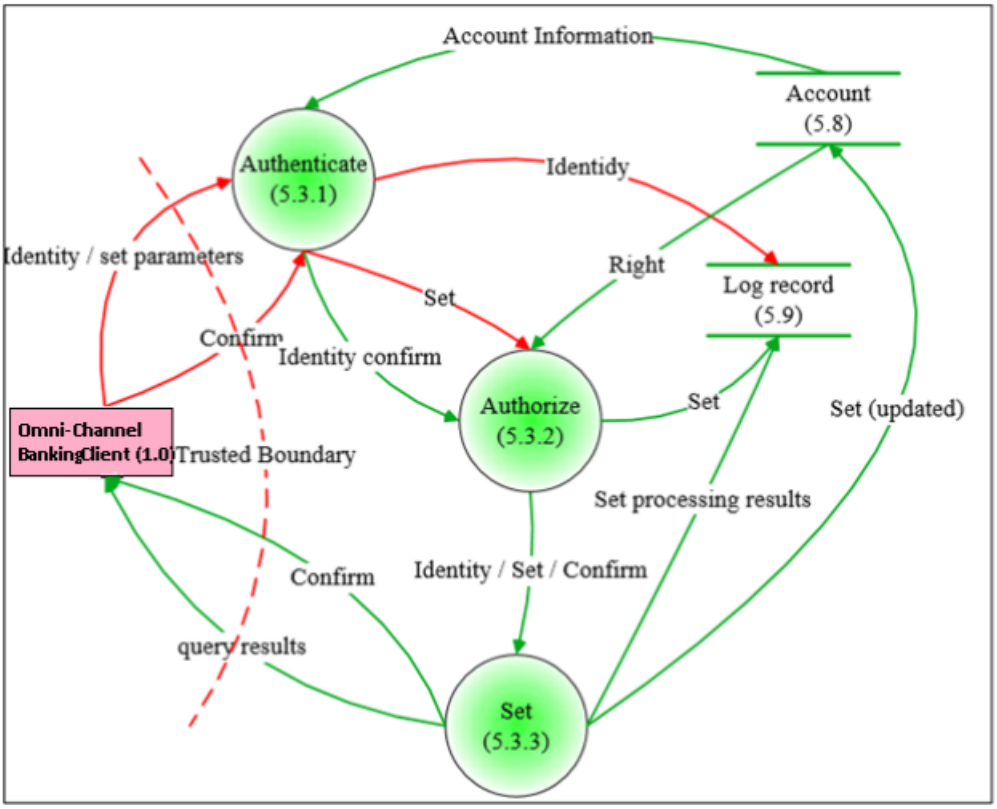

Figure 4.3: Set Type of Transaction

Core banking system client should be able to perform electronic payment(5.5.2) process and got back response.

Figure 4.6 shows the cash changing process. Client should perform Authentication(5.6.1), Authorization(5.6.2) and Transaction(5.6.3) process with the uses of Account, Log record Data store. Transaction process send request for transaction to Core banking system. It gives result to Transaction, Through it final result send to OCB client that transaction request is confirm or deny.

Management stuff perform system management process as shown in Figure 4.7. For that Management stuff send Identity or Operating parameter to system for Authentication(5.7.1) process, now request pass to Authorization(5.7.2) process. So they can able now to perform operation For manage logs and Account data store. Management stuff perform various task on system, all the performed task on system should be logged.

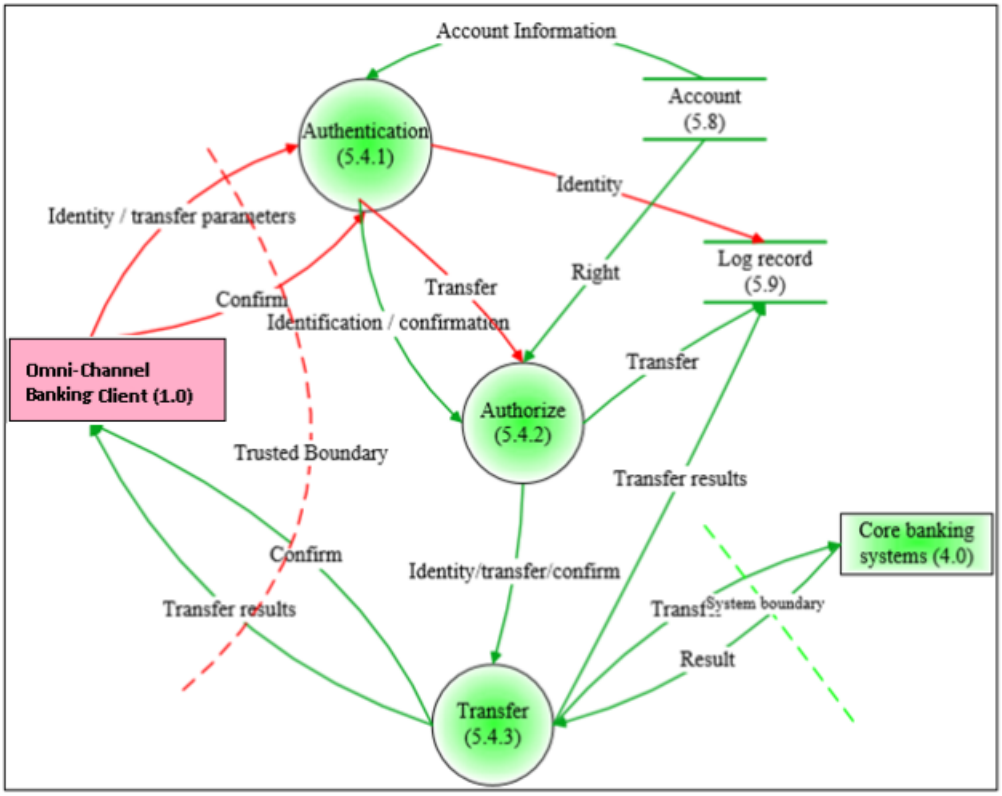

Figure 4.4: Transfer Type of Transaction

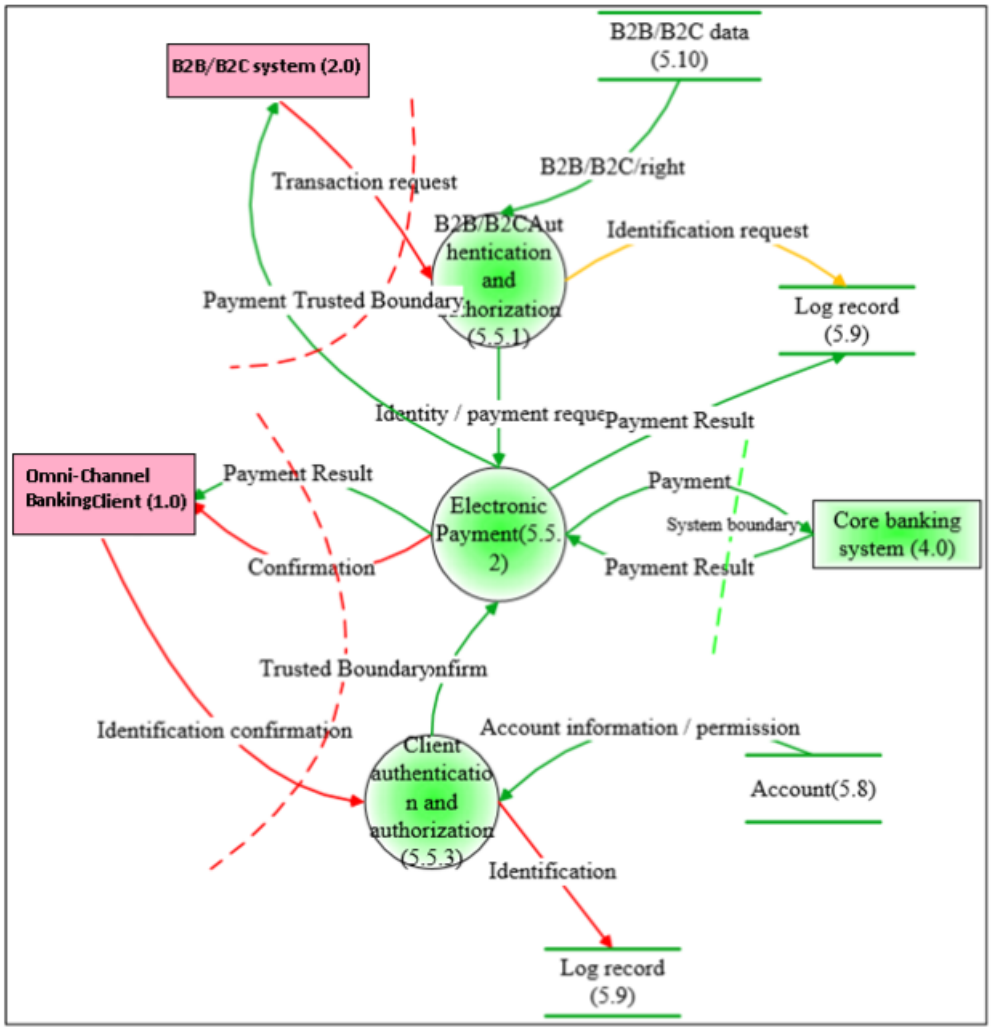

Figure 4.5: Electronic Payment

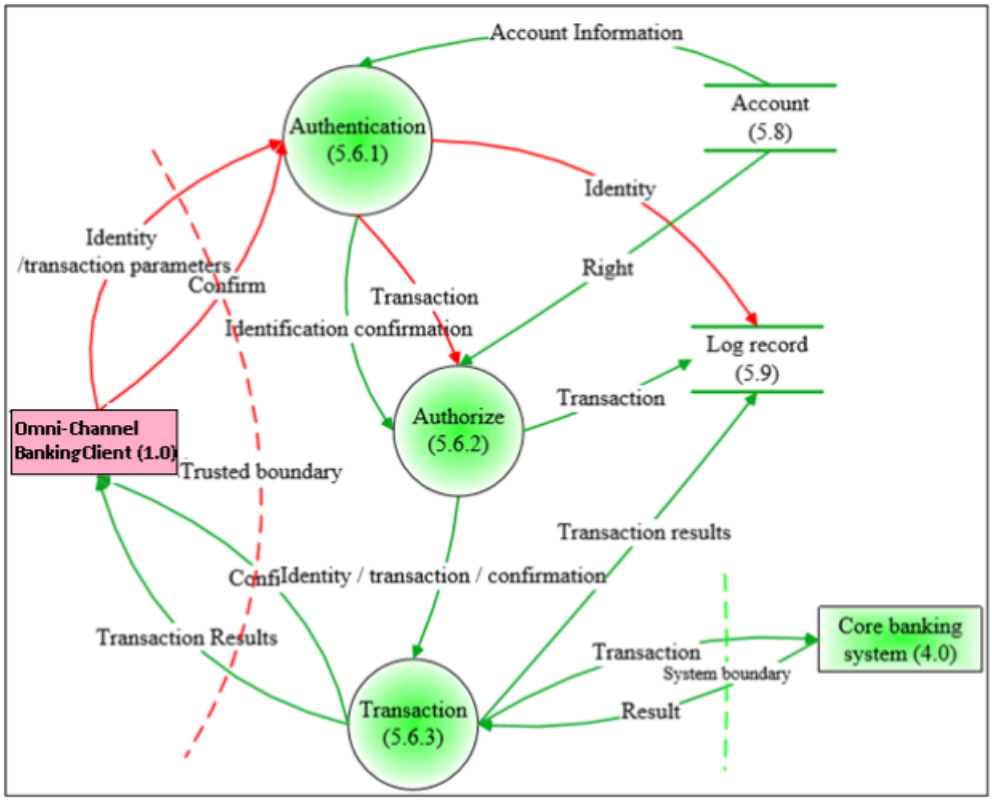

Figure 4.6: Cash Changing Process

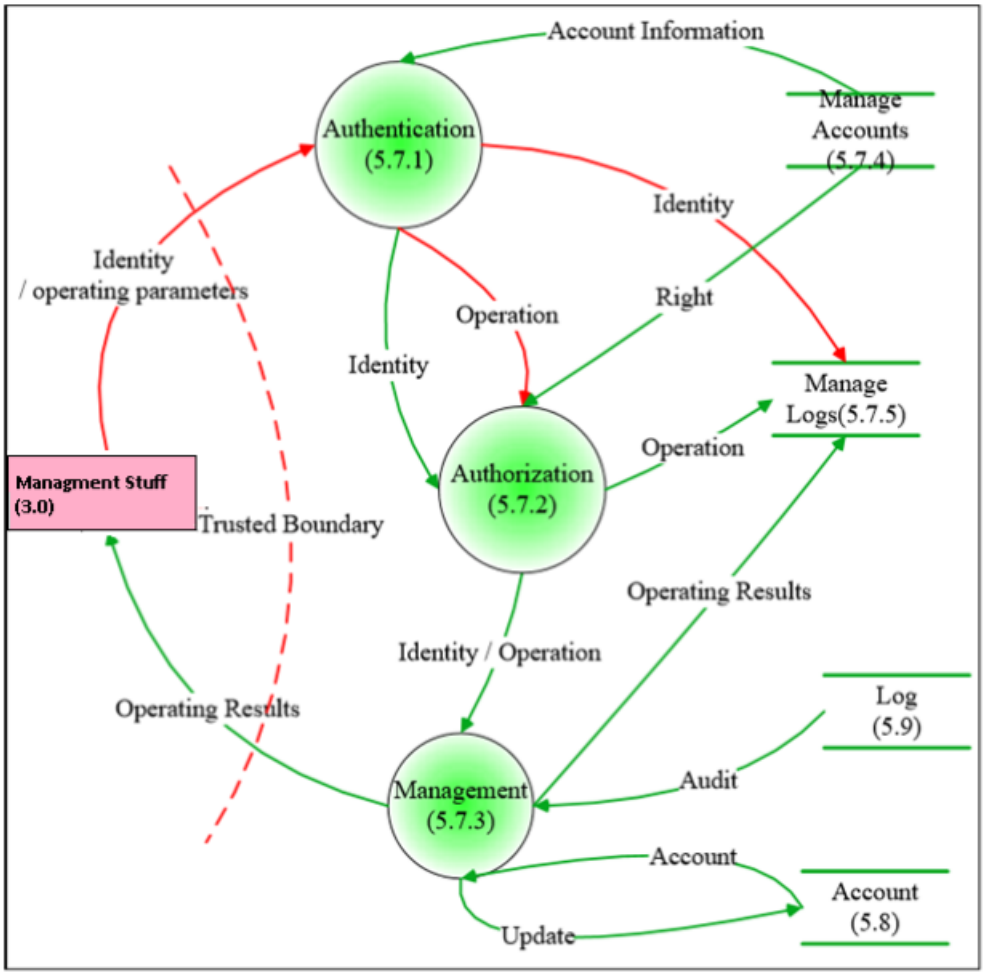

Figure 4.7: System Management

4.2 Identifying Threats

Table 4.1 shows the list of identified threats, which is categorized into the following Properties: authentication, authorization, Session Input Validation, Data Protection, and Availability. Also they gives details about the Threat and their Countermeasure which are present at that property in system.

Properties	Threat ID	Threat	Countermeasure	STRIDE
Authentication	T1.1	Illegitimate requesters could get access to confidential resources while pretending to be legitimate requesters, thus stealing legitimate requesters identity and data, executing unauthorized actions	If access to functions or data provided or managed by the software is subject to access control, users need to be identified and successfully authenticated before such access is granted by the software. To achieve this, the following requirements need to be fulfilled: 1.Software shall be able to differentiate users by different User IDs 2.User ID/Password 3.Single sign-on 4.Software supporting multi-tenancy shall be able to differentiate User IDs and configured identity management and authentication options	S
Authentication	T1.2	Attackers could exploit a weak implementation or configuration of username and password authentication, like password complexity or password storage.	Providing User ID/Password authentication securely. The following requirements shall be met: 1.User ID: Do not map some characters in the User ID to the same one if this can lead to ambiguities. 2.Password: Passwords shall always be case sensitive, minimum password length, maximum password length, whether lower case and upper case characters, digits and/or special characters are required in passwords, number of previously used passwords not allowed as a new password, number of subsequently failed logon attempts until the user account gets locked 3. Transfer: It shall be possible to use a secure channel for transmitting the User ID and password information	S
Authorization	T2.1	Users or attackers can get access to unauthorized data or functions to steal, modify and/or delete confidential data.	OCB have to provide the capability to separate the authorizations for different entitlement users.	T, I, E
Authorization	T2.2	Attackers could get access to these replicated data that are sometimes less protected and get unauthorized access to the confidential data.	OCB should enforce authorizations for the access to replicated data.	T, I, E
Authorization	T2.3	Attackers could exploit misconfiguration between the different authorization system on the separate access paths to gain illegitimate access to confidential data.	OCB should enforce consistent authorization checks for all possible access paths.	T, I, E
Session	T3.1	Attackers could view/steal, tamper or delete these cookies and therefore impersonate the legitimate users or view, modify or delete their confidential data.	Protect security sensitive cookies	S, R
Session	T3.2	Attackers could exploit XSRF vulnerabilities to submit requests (retrieve confidential data, modify confidential data) on behalf of the legitimate user and remain unnoticeable by any log mechanism.	No Cross-Site Request Forgery vulnerabilities.	S, R
Data Protection	T4.1	The unavailability of means to read access log to sensitive personal data leads to violation of data protection and privacy regulations.	Access to the log records shall be restricted by adequate authorizations.	R
Input Validation	T5.1	first attacks Attackers test to hack into a system are SQL Injections. Attackers using SQL Injection could steal, modify or delete confidential data.	No database query injection vulnerabilities.	S, T, R, I, D, E
Input Validation	T5.2	Attackers can launch attacks against vital elements to gain access to confidential data or block resources	Provide more security into logging page, Try to give security against CSRF, XSS, Social Engineering, etc	T, R
Input Validation	T5.3	An attacker could upload files which contain viruses or MIME-type sniffing or exhibit other malicious behavior when opened for processing.	Protect upload, download and display functions of untrusted files against MIME-type sniffing and virus attacks.	S, T, R, I, D, E
Availability	T6.1	Attackers could interrupt the functionality or services partially or completely offered by your application and render access to confidential data infeasible. In some cases attackers could delete confidential data completely from your application.	Provide Protect against Denial of Service	D

Table 4.1: Identifying Threats at Different Properties

4.3 Omni Channel Banking External Entity Threat Analysis

Here, our aim is to perform threat analysis of the OCB which include external entity (Omni channel banking client, B2B/B2C system, and Management staff). Earlier we represent the threat definition and their countermeasure.In this section we present External entity threat analysis, as shown in Table 4.2, 4.3.

STRIDE	Threat ID	Omni channel banking Client	B2B/B2C system	Management staff
Spoofing	**T1.1, T1.2, T3.1, T3.2, T5.1, T5.3**	S1. Fake customer character. S1.1. Illegally got endorsement. S1.1.1. Legal declarations acquired by the assailant. S1.1.2. Forged testament. S1.2. Certification unsecure. S1.2.1. Lack of confirmation instruments. S1.2.2. Certification isn't adequate. S1.2.3. Server's validation defenselessness, which can be circumvent. S1.2.4. Authentication calculation un-secure driving Man-in-the-Middle assault. S1.2.5. Certification process is re-executed. S1.2.6. Passwords be broken. S1.3. Password Security. S1.3.1. Password quality is deficient, can be split. S1.3.2. Default watchword is shaky. S1.3.3. Password stockpiling isn't secure. S1.4. Brute power. S1.4.1. Lack of system to oppose savage power. S1.4.2. Mechanism of protection from animal power can be skirted. S1.5. Session component isn't flawless. S1.5.1. Lack of session timeout instrument. S1.5.2. Lack of session state examination. S2. Counterfeit customer personality correspondence. S2.1. Malware reproduce console activities. S2.2. Malware reenact customer sending message. S2.3. Malware counterfeit client's task	S1. B2B/B2C is an extortion webpage. S1.1. The server website to URL. S1.2. Domain satirizing. S1.3. Content caricaturing. S1.4. Framework is inserted in a site. S1.5. ARP ridiculing seized course back to the false website data. S2. B2B/B2C is a phony webpage. S2.1. Illegally acquire endorsement. S2.1.1. B2B/B2C lawful declaration got by aggressor. S2.1.2. B2B/B2C testament is phony. S2.2. B2B/B2C verification isn't secure. S2.2.1. Not for the confirmation of B2B/B2C. S2.2.2. B2B/B2C validation isn't adequate	S1. Forged manager character. S1.1. Obtain declaration unlawfully. S1.1.1. Administrators lawful accreditation got by attacker. S1.1.2. Forged endorsement. S1.2. Authentication is unsecure. S1.2.1. Administrator confirmation is inadequate. S1.2.2. No Administration validation. S2. Management have is fake activity subsequent to being attacked.
Tempering	**T2.1, T2.2, T2.3, T5.1, T5.2, T5.3**	T1. Client is installed noxious program. T1.1. Malware altered the client ask for information or the server returns information. T1.2. The information of malware altered client input. T1.3. Malware change program memory. T1.4. Malware to adjust the message sent or got. T1.5. Malware show information by client activities in the interface. T1.6. Malware adjust console input	T1. B2B/B2C site is installed malware. T2. B2B/B2C site is controlled by assailant	
Repudiation	**T3.1, T3.2, T4.1, T5.1, T5.2, T5.3**	R1. Users deny completed exchanges. R1.1. Lack of exchange signature component. R1.2. Log record isn't flawless. R1.2.1. No log records. R1.2.2. Log records deficient.	R1. B2B/B2C business party deny did exchange. R1.1. No legitimate mark. R1.2. Log record isn't impeccable. R1.2.1. No log records. R1.2.2. Log records deficient	R1. Executives deny activity. R1.1. No substantial mark. R1.2. record isn't flawless. R1.2.1. No log records. R1.2.2. Log records deficient.
Information Disclosure	**T2.1, T2.2, T2.3, T5.1, T5.3**	I1. Malware to take client touchy data, for example, passwords, declaration, and information. I1.1. Malware to take passwords and other delicate data by means of the console record. I1.2. Malware to get delicate data, for example, client secret key by screen captures. I1.3. Malware to take the program information in memory. I2. Sensitive data without secure taking care. I2.1. Sensitive data put away in the neighborhood prompt data spillage. I2.2. Encryption scratch is put away on the customer. I2.3. Used customer brief documents does not be erased in time. I3. Fake site to cheat client input. I3.1. User is angled I4. Security component isn't immaculate. I4.1. No message security instrument. I4.2. Weak message security system. I4.3. No channel security component.		

Table 4.2: OCB External Entity Threat Analysis-a

STRIDE	Threat ID	Omni channel banking Client	B2B/B2C system	Management staff
Denial of Service	**T5.1, T5.3, T6.1**	D1. Channel over-burden, driving log handling hang or crash. D1.1. Abnormal parameters prompt countless of server memory or CPU. D1.2. Abnormal parameters make the server a business is to hang or crash. D1.3. Multiple simultaneous activities cause the server isn't reacting. D1.4. SYN FLOOD/HTTP FLOOD assault. D1.5. Large number of system parcel blocking system. D2. Undermine message trustworthiness	D1. Channel over-burden, driving log handling hang or crash. D1.1. Abnormal parameters prompt an expansive number of utilization of server memory or CPU. D1.2. Abnormal parameters make the server a business is to hang or crash. D1.3. numerous simultaneous activities cause the server isn't reacting. D1.4. SYN FLOOD/HTTP FLOOD assault. D1.5. extensive number of system parcel blocking system. D2. disappointment message respectability	D1. Channel over-burden, prompting hang or crash log preparing. D1.1. Abnormal parameters prompt countless of server memory or CPU. D1.2. Abnormal parameters make the server a business is to hang or crash. D1.3. numerous simultaneous activities cause the server isn't reacting. D1.4. SYN FLOOD/HTTP FLOOD assault. D1.5. Large number of system bundle blocking system. D2. Server information has physical harm or decimation. D3. Host flimsy, which causes information blunders
Elevation of Privilege	**T2.1, T2.2, T2.3, T5.1, T5.3**	E1. Client security vulnerabilities. E1.1. Client control of security vulnerabilities, prompting hanging horse. E1.2. Client framework part driver of security vulnerabilities		

Table 4.3: OCB External Entity Threat Analysis-b

4.4 Summary

By thinking about dangers of different classes for single component in DFD, STRIDE give the distinguishing proof of threats inside the application. In this section we did information stream investigation of OCB framework, likewise performed examination of whether every datum stream and it's related resource data is defenseless against a Spoofing, Tempering, Repudiation, Information Disclosure, Denial of service and Elevation of privilege dangers. Likewise drill down the influenced properties. We did threat investigation of the Omni channel banking application including outside substance, the information stream and the information stockpiling incorporating into business tasks, as opposed to isolate examination of activity.

Chapter 5

Analysis of the OCB Threat based on Attack Tree

For analyzing threats into omni channel banking here we implement attack tree. First, we determined high level threat, than decompose this threat into intermediate objectives. Using of that objective we decomposed into individual attack scenario. Create the Systems one attack tree is impossible, because present the all attack scenario on one tree is not possible. So, we consider one attack condition and then decompose into attack tree, try to cover all possible attacks at that node.

Following attack tree is constructed here,

- Attacker gain access to OCB
- Attacker gain access to users personal account
- Attacker obtain Admin Privilege

First we analysis the situation when attacker gain access into Omni channel banking system. So, in Figure 5.1 root node represent the attacker is to gain access into OCB. And child node represent conditions to achieve the root goal.

Figure 5.2 present the scenario in which attackers ultimate goal is to gain access to users account. The child node represents sub-goals necessary to achieve the root goal and attackers actions which they perform. Which condition and attacks are used by attacker to obtain admin privileges are shown in Figure 5.3.

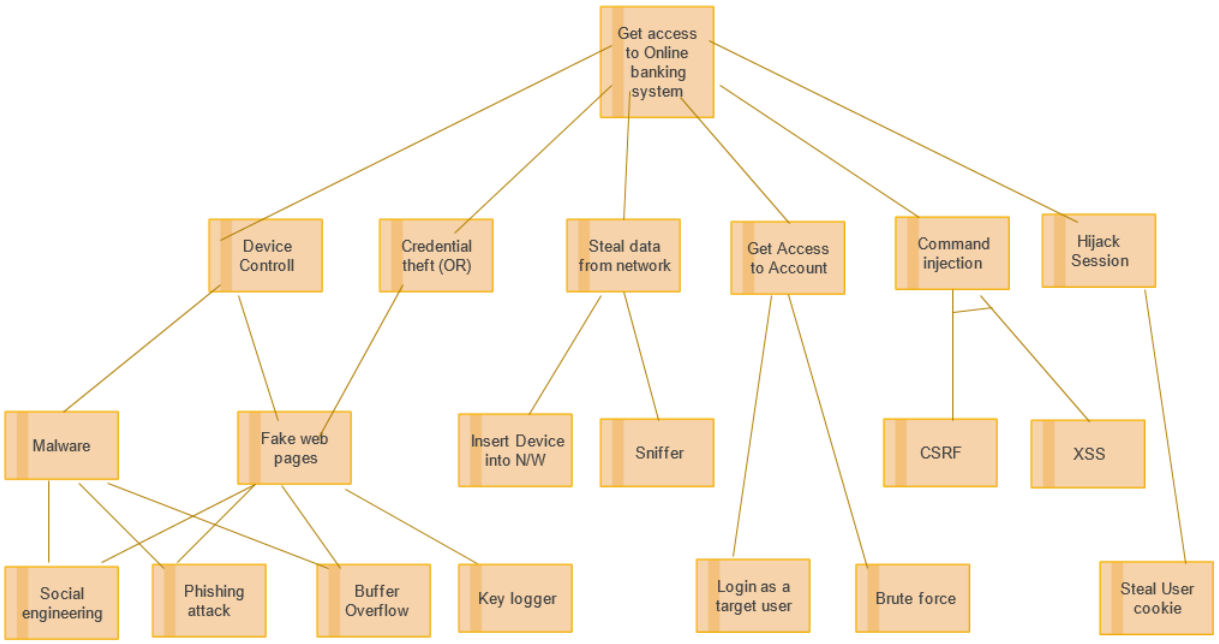

Figure 5.1: Attack Tree When Attacker Gain Access to OCB

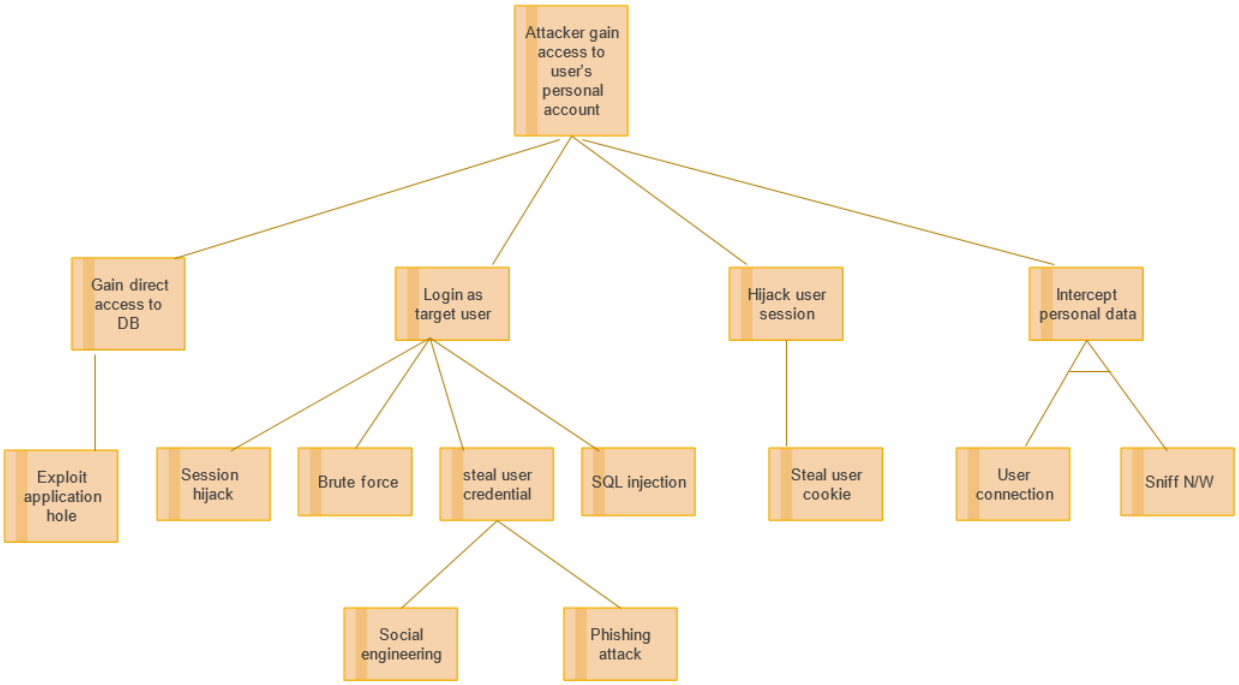

Figure 5.2: Attack Tree When Attacker Gain Access to Client Account

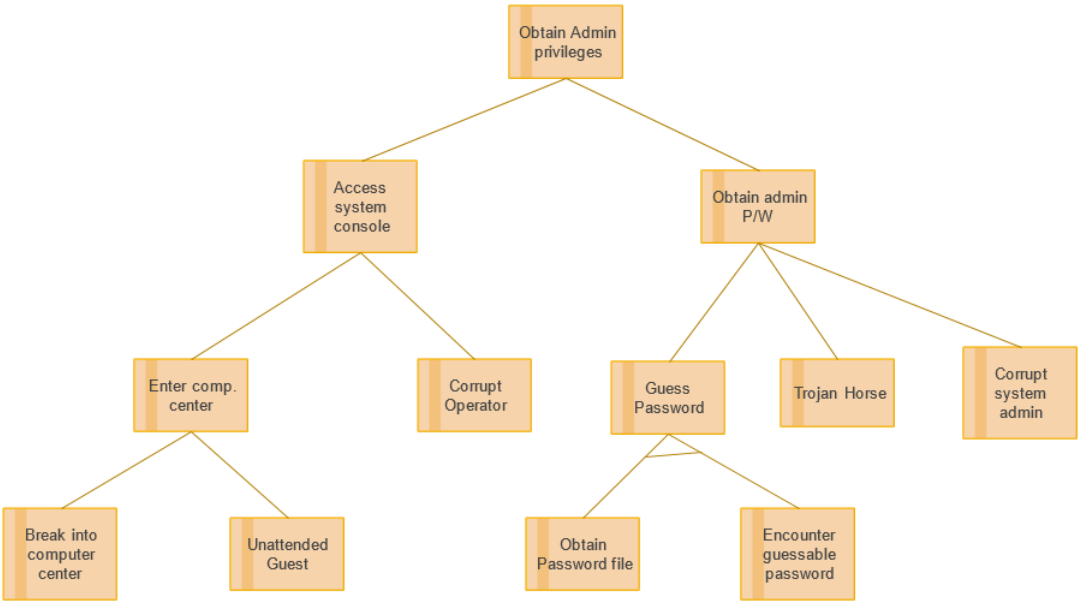

Figure 5.3: Attack Tree When Attacker Obtain Admin Privilege

5.1 Summary

Attack tree [5, 11] presents the formal and systematic method for threat analysis of system threat. All the threat with single attack tree to system modelling is not realistic because the attack tree will be very large. Based on the above reasons, this chapter include attack tree when attacker gain access to OCB, Attacker gain access to user's personal account and attacker obtain admin privilege. This attack trees created using of existence attack tree of system so if existence system didn't include new attack than our attack tree goes fail at that particular attack node.

Chapter 6

Implementing Attack Tree Using STRIDE Model

Analysis for system threat is carried out in systematic manner by Attack Tree. In real, the structuring of all the threats in system becomes complex. So, all the related threats using a single attack tree in a system is not possible, as the attack tree will become very huge. Taking into consideration of threats, the thesis monitors the STRIDE threat model and the analysis using attack tree altogether. Focusing on STRIDE mode, the system is divide into five parts, S, T, R, I, D, E, so that the threats are targeted in every corner. The analysis is done using Attack Tree Threat on every module with different category. In this case, complexity is reduced.

6.1 Logic Relationship of Attack Tree

The Attack Tree has a hierarchical structure to model threats by keep threat action in organized manner, that is, the structure is based on Data structure where the nodes are connected in hierarchical manner with directional edges. It is mainly represented by text, and the relationship among the attack trees is given by AND, OR.

6.2 Implementing Attack Tree Using STRIDE Model

Analysis of every possible threat in the Omnichannel banking system and decomposition of the attack way have the following methods which are mentioned below:

1. Omnichannel Banking security threats are classified with the help of STRIDE threat model.

2. Decomposition is done gradually when the mode of attack is recognized from which the middle layer nodes are formed. This comes under top- done approach in attack mode.

3. The child nodes are checked to verify the upcoming decomposition, in this case child nodes are made as a current target. The process is repeated so that the modules can be broken into smaller parts.

4. The further case is that if the node involved cannot be further decomposed, it is terminated from the decomposition process. The inverted attack tree is built where every child node becomes independent and can access the components.

This chapter only gives omnichannel banking client fake attack tree as an example Figure 6.1.

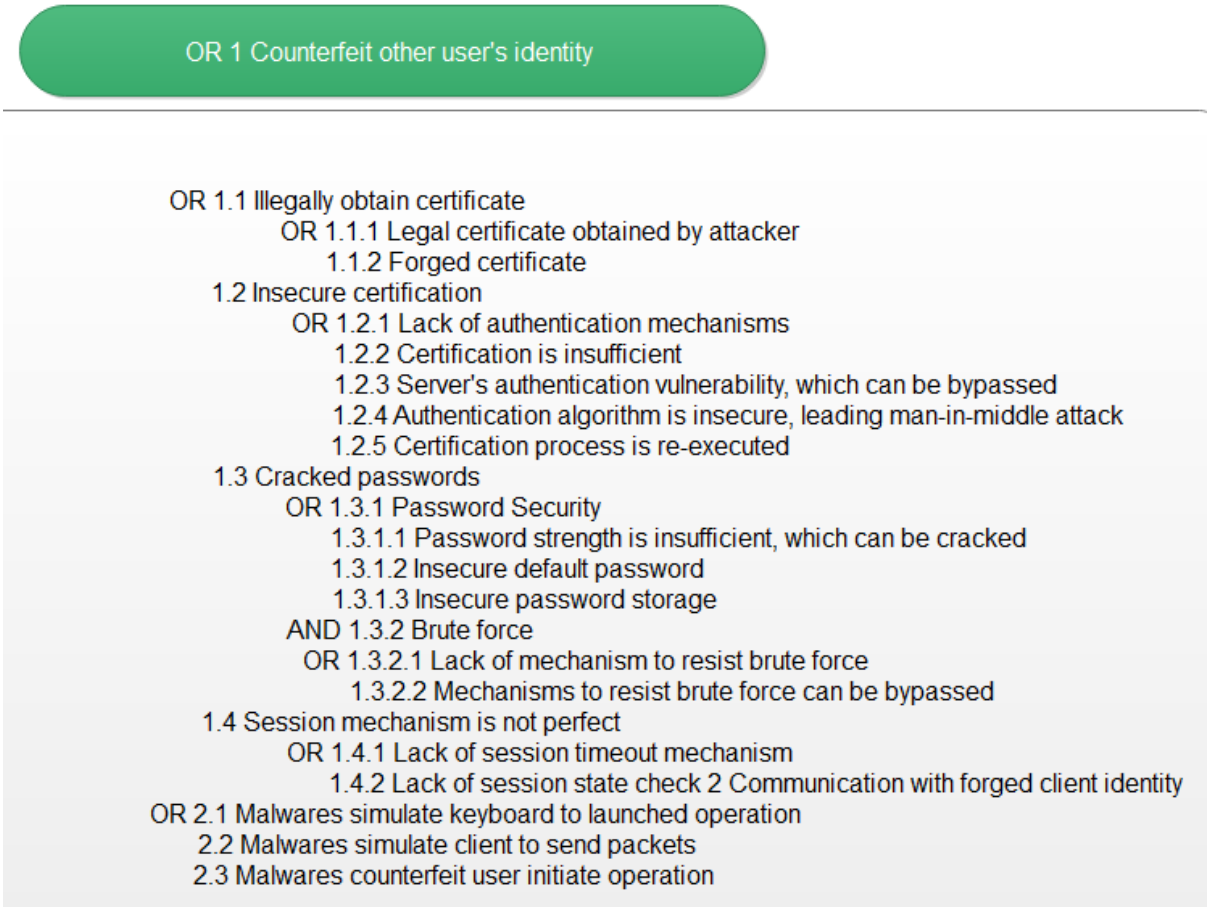

Figure 6.1: OCB Client Fake Attack Tree

For Omnichannel Banking system, the Attack Tree consist of all the threats in a system. Any related threat is a risk to the Omnichannel Banking system. The analysis

for external entity is shown in Table 4.2 and 4.3.

6.3 Summary

So, basically an Attack tree is a threat recognizing tool with the representation of tree, so it helps to reuse and structure the system. The previous studies show the lack of systematic and holistic use of Attack Tree threat analysis, which leads to difficulty in carrying out efficiency of Attack Tree. The Chapter summarizes STRIDE threat model with the help of attack tree analysis, greatly reducing the attack tree constructed complexity, which makes it less difficult to use and maintain the attack tree. Through the threat analysis of omnichannel banking system, the Attack Tree method well describes the omnichannel banking system security threats, and provides guidance for system security analysis and evaluation.

Chapter 7

CSRF Attack and OData

7.1 CSRF Attack

The attack in which fraud happening more and more, called cross site request forgery attack [8]. These is common attack in now a day. Most of the time CSRF attack is happening in web application and this will make the website more harmful for all the users who are using that website occasionally.

Take one example of CSRF attack. One user wants to do some transaction on the website. The website is SBI banking website. After filling the form of the transaction, loaded page will ask you to choose option from debit card or credit card. After choosing the option and enter the amount that user want to send. After clicking on the submit button the site will redirect to another website which is same as a bank website. User assume that this is a right website but that is a wrong website. Money is reached to any third-party account not in the actual receiver's account. These attack is called as a CSRF attack.

Simplest way to decline these attack is that, generate some random number which contain unpredictable string in which that number same throughout the whole session [6]. When every request is made, that number is checked internally and if is not same than session is logged out otherwise it will continue. In these manner, any developer can prevent the CSRF attack and generated number is called x-csrf-token. These x-csrf Tokenkey will generate differently at every logged in session and stay unique till session logged out. In Figure 7.1 csrf attack overview is there.

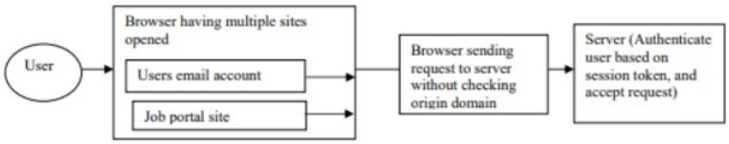

Figure 7.1: CSRF Attack Scenario [5]

7.2 OData

Apache Olingo is one way to implement the Open Data Protocol (OData). Apache Olingo itself is a java library [8]. With the help of Java library Apache Olingo create some OData services. Apache Olingo have two aspects which is server and client respectively. Now a day, Apache Olingo supports OData 2.0, but newer OData 4.0 which is also supported by Apache Olingo. Expose the JPA entity as an OData services with the help of Apache Olingo. Consuming the OData services in the application by Application's UI layer which is directly used by the end-user. Application data received through OData. Services related to OData represent application business objects and their relationship to other lists of business object in third party applications. Any application consumes OData web services to send the data to collaboration groups.

7.2.1 Life Cycle of OData

Life cycle of OData are given into following steps,

1. Activation of any service through OData
2. Maintaining of any service through OData
3. Maintaining of any services and different models will helpful to cleaned up data called metadata of the application. It will clean from the cache.
4. Any application which contain REST for using HTTP request. It will post the data in any application. It will create, delete, update, read operation called CRUD operation.
5. Web Services and Remote Procedure Call (RPC) are convention for any lightweight RESTful application.

7.2.2 Apaches Olingo's Services for OData

Following are the some Apache Olingo's services for OData.

1. EdmProvider: It is entity data model provider service. which is delivers an abstract definitions for services. This service is used to redeem complete structure information in order to build the metadata document and the service document.

2. DataProvider: Apache Olingo library is included in the data provider to connect any data source.It is more helpful with the static type of data. Data provider is mainly used for establishing a connection with the database. Also it is helping for build an application with no constraint.

3. Processor: Every processor invokes when request arrives. It receives, de-serializes and validating every request. After validating it will fetch some requested data. And send the response of the data to the client in the serialize form.

7.3 Software Used for OData Request and Response

Following are the list of software to be used for checking OData request and response,

1. POSTMAN

2. JMETER

7.3.1 POSTMAN

If any application want to make some API request therefore application name called postman which is extension in the google chrome application or developer can download the postman application [8]. Authorization is required make any API request from postman. Testing web services through the postman application which make powerful HTTP request. With the help of postman any developer can easily develop, test and making document for any simple HTTP request or complex HTTP request. JavaScript is also used to set the environment variable or global variable. The stored value from the variable is used into further request. It will make chained throughout the whole script.

Developer can also import a file from outside the postman application which contain JSON or CSV format. Testing the results from the postman can see through the collection

runner. The main thing in postman to validate any API request which is made by the developer and it will give the response into the JSON or XML format. The JSON or XML response is in readable format and that response is come within the application. Postman can be run through command line, too. The command line tool for postman is called newman. Postman is far better than any API testing tool. It is part of any API life cycle.

Postman is used for making OData request from the HTTP request. OData request can be made as a URL parameter and the HTTP request may contain GET, POST, DELETE, PUT etc. Table 7.1 shows operator and their values into OData through the Postman application. Sometimes pre- request script can be made through the request. Test cases and assertion will make the OData request more powerful. Passing the request through body and request can be made through header parameter in the postman application.

Table 7.2 have some request which can help in the searching request through the postman application.

Operator	**Usage**
Equal	$filter = City eq ahmedabad
Not equal	$filter = City ne banglore
Greater than	$filter = Salary gt 5000
Less than	$filter = Salary lt 1000
Greater than or equal	$filter = Salary ge 5000
Less than or equal	$filter = Salary le 1000
Logical AND	$filter = Salary ge 5000 and Salary le 1000
Logical OR	$filter = Salary ge 5000 or Salary le 1000
Logical NOT	not endswith (Description ,'samy')
Addition	$filter = Price add 5 gt 10
Multiplication	$filter = Price mul 5 gt 10
Subtraction	$filter = Price sub 5 gt 10
Division	$filter = Price div 5 gt 10
Modulo	$filter = Price mod 5 eq 0
Precedence grouping	$filter = (Price add 5) gt 10

Table 7.1: Operator on OData

7.3.2 JMETER

Jmeter is mainly used for testing purpose. It will test the load testing and performance testing for any java application. Cloud OData URL is also being checked by the Jmeter

Request	Description
$filter	It will lead to strong search criteria
$expand	Expand some field in the response
$action	Perform wizard operation
$skip	Skip some results
$top	Show top results
$inline	Show pages

Table 7.2: Searching Request on OData

tool. Jmeter have different types of testing plan, functions, regular expressions, json extractor, bean shell timer, listener. The Jmeter software is developed by the Apache. HTTP protocol is used by the Jmeter to listen any OData request. Response can be in JSON format and Request can be made through JSON, field parameter, file included etc. Figure 20 defining the work flow of the Jmeter.

From Figure 7.2, we can say that group of users will be make as an input and it will generate some output in the report or in some graph manner or may generate some report for that.

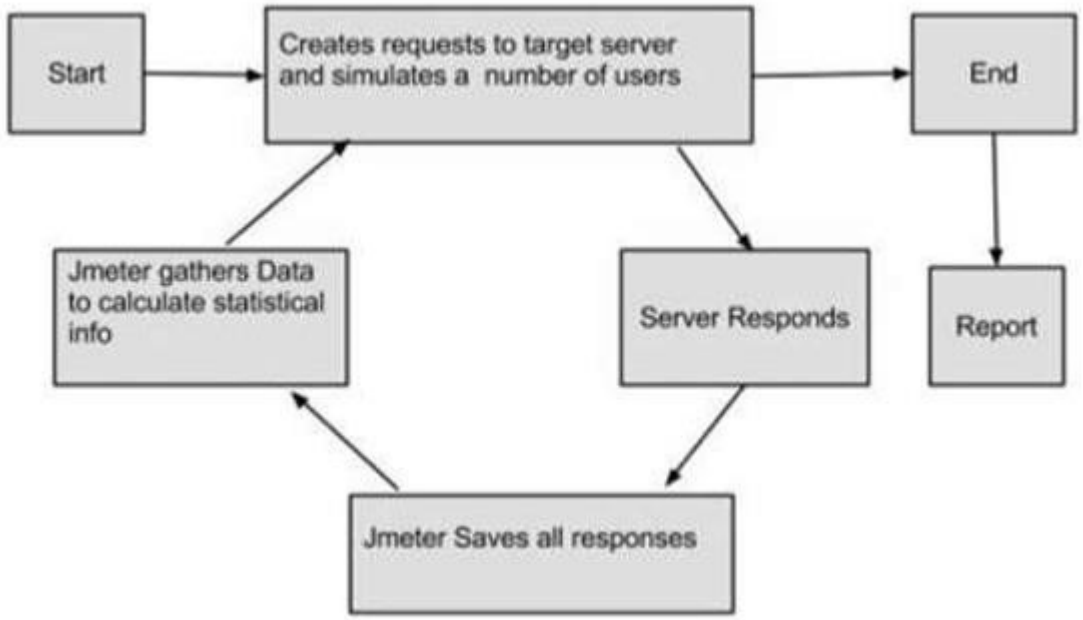

Figure 7.2: Working of JMETER

7.4 Difference Between Apache JMETER and POSTMAN

Main difference between Jmeter and postman is number of user as input. Jmeter takes multiple user as an input and postman takes 1 user as an input at a time [5]. Second major difference between Jmeter and postman is regarding test case and storage of environment variable. In the postman, environment variable is stored through writing a test case for that but in Jmeter test case cannot be written but value can be stored for further request via regular expression or json extractor.

7.5 Summary

This chapter given introduction about CSRF attack and OData (Open Data Protocol). Here also introduced the technique to avoid CSRF attack happaning to application. Now days many techniques are using to prevent CSRF attack. But here we introduced new concept of preventing CSRF attack through OData. OData generates X-CSRF TokenKey, So User's session are more secured now. Also introduced list of software which are used to check OData request and Response. POSTMAN tool is mostly commonly usable tool for OData. But when application want to check multiple request and response, that time JEMETER is used.

Chapter 8

OData Through Implementing X-CSRF TokenKey

For implementing X-CSRF TokenKey into OCB, first OCB have to implement OData into application. After implementing OData into application, Enabling it for generating X-CSRF TokenKey. We also check session key and OData request-response through POSTMAN tool.

OData is helpful for accessing and retrieving the data from the database in the proper format. Following steps to be followed for implementing OData into Omnichannel Banking(OCB). Also using of fourth step we can genrate X-CSRF TokenKey into Application. Apply following code changes to web.xml file. So, Application can generate secured session key through OData.

1. Enabling OData through code
2. Get the data entity model from file into java code.
3. Regestring Odata into application using following code.

   view.registerOperations (Class name, null)
4. Apply changes into web.xml file for implementing X-CSRF TokenKey.

OData URLs can contain sensitive data that is assumed to be protected using HTTPS. A CSRF token-based protection has been introduced for all modifying requests. For that first get a valid X-CSRF token from server than Send the actual HTTP post request to

server by appending the X-CSRF token in request header eld which is got from previous step.

8.1 Checking Session Key from Front-Side

First load the application into Google browser. than go to developer toll by pressing F12 key from keyboard. it redirect to Google plugin developer tool. from where we can check session key after user logging into application. So as shown in Figure 8.1 at logging time application generate TokenKey and X-CSRF TokenKey. which is consistence at user's session. if any changes detected into TokenKey and X-CSRF TokenKey, then user automatically logged out from application. user can't available for further operation. every session unique TokenKey and X-CSRF TokenKey are generated. but at one session it is consistence. So as shown in Figure 8.2, when user Transfer the amount that time key value are checked and if any moderation are not find from TokenKey and X-CSRF TokenKey then user can available for doing transfer amount to other account.

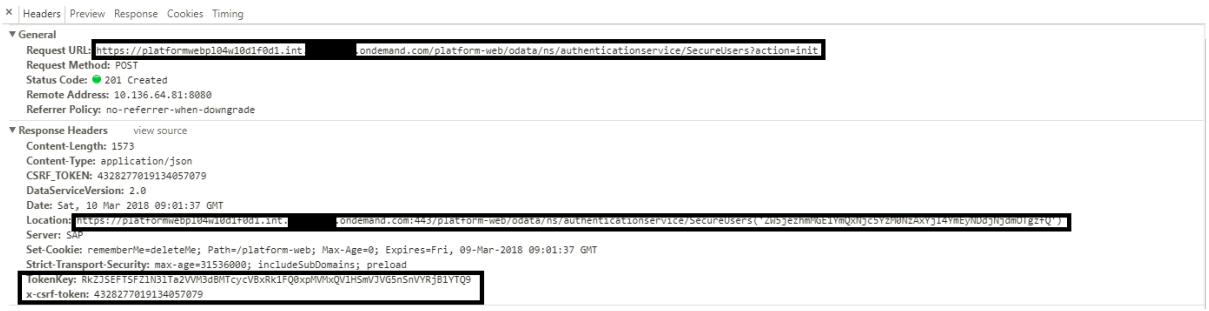

Figure 8.1: Session Key at Logging page

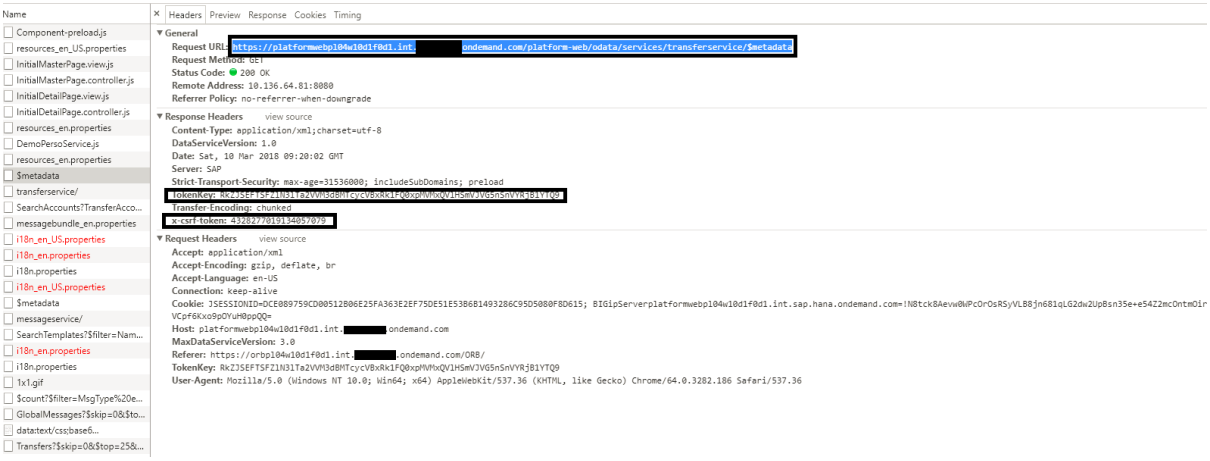

Figure 8.2: Session Key at Transfer page

8.2 Use of POSTMAN for OData

Here we used POSTMAN tool for checking OData request and response. For that first load the application link into Postman. As shown in Figure 8.3 Postman gives response to logging request. in environment variable we have to stored environment link. As shown in Figure 8.4 header through Postman stored TokenKey for a user's session. For that first gives the header details and write content into body and Test given into Figure 8.5, then send the request for response. Define the test scenario into Test field. It is used for applying condition to request. Checked the response for link. if status is 202 Passes then request passed successfully to application server. here, our link passed through security so we have PASS status, As shown in Figure 8.6.

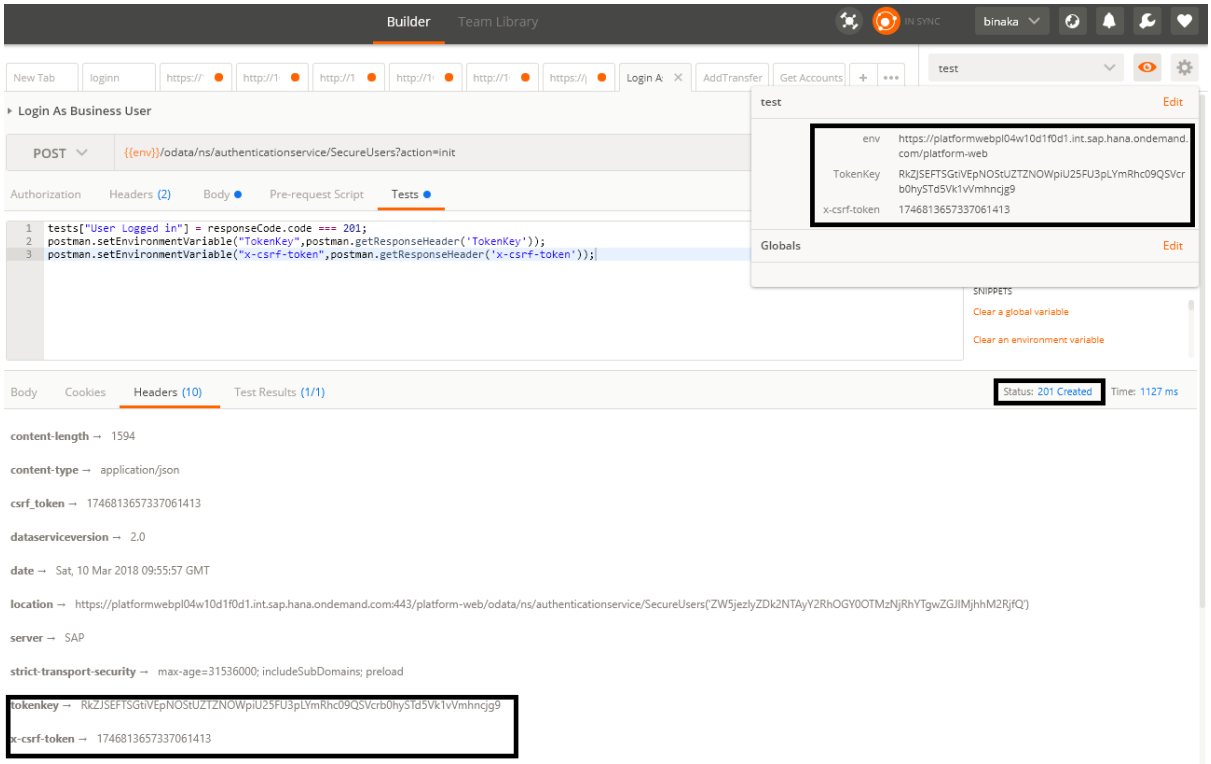

Figure 8.3: Logging Request and Response

Figure 8.4: OData Request and Header

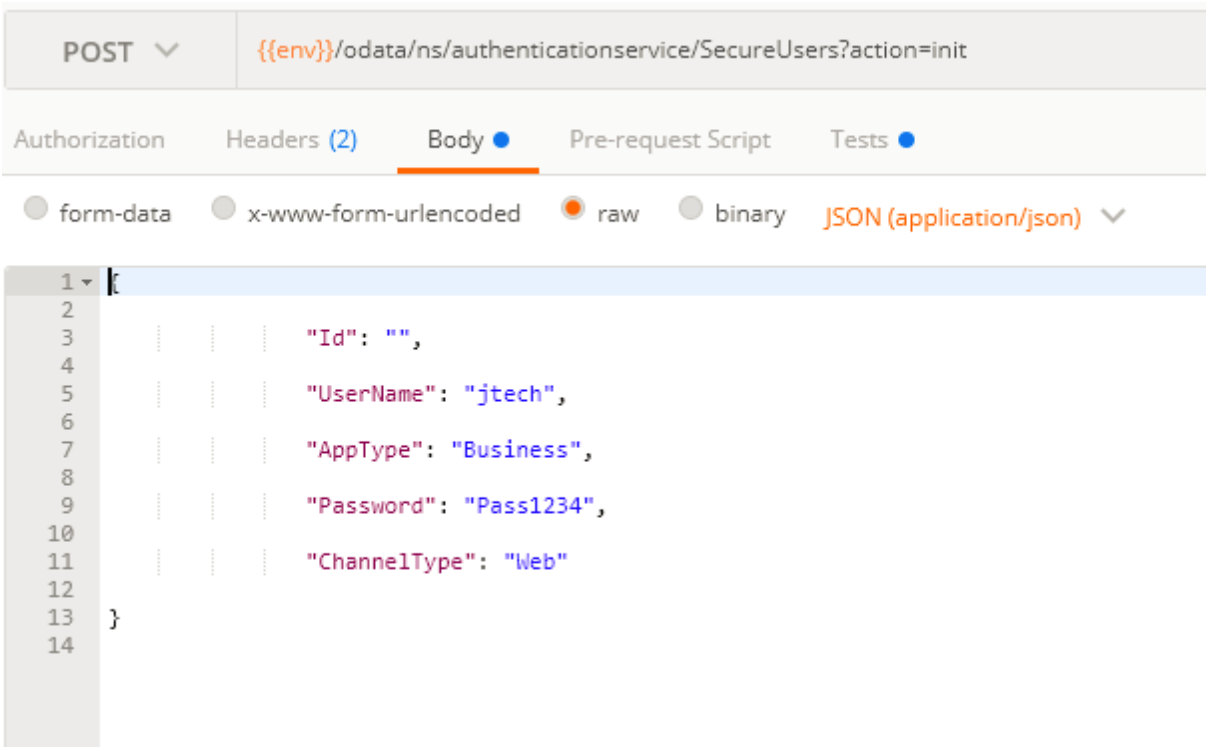

Figure 8.5: JSON as Body Request

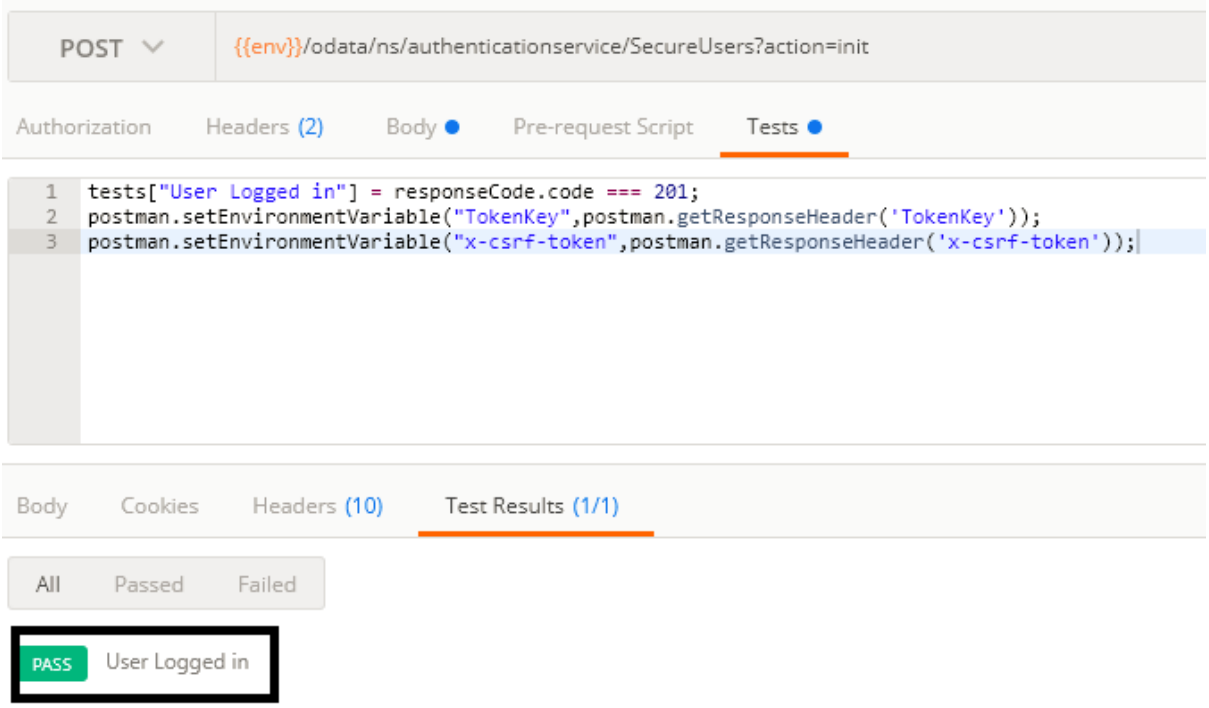

Figure 8.6: Logging Test Case and Result

For next request user have to stored TokenKey and x-csrf-token variable into Header as shown in Figure 8.7. Stored x-csrf-token and TokenKey is passed in next request in double curly braces {{}} and it is passes through the header. Then write body for

next request and also create test for Transfer. Figure 8.8 shows the input of Test and output of Test Results, which are passed here. Here our application matches environment variable for all POST request and after satisfying Test case it gives the Positive response. Figure 8.9 shows response of transfer request. Where Passed Token key are matches with request, So we got Passed status. If new request has applied any changes to TokenKey then request goes wrong and got the Failed output. So, OData checking every time TokenKey when any request from user got into session.

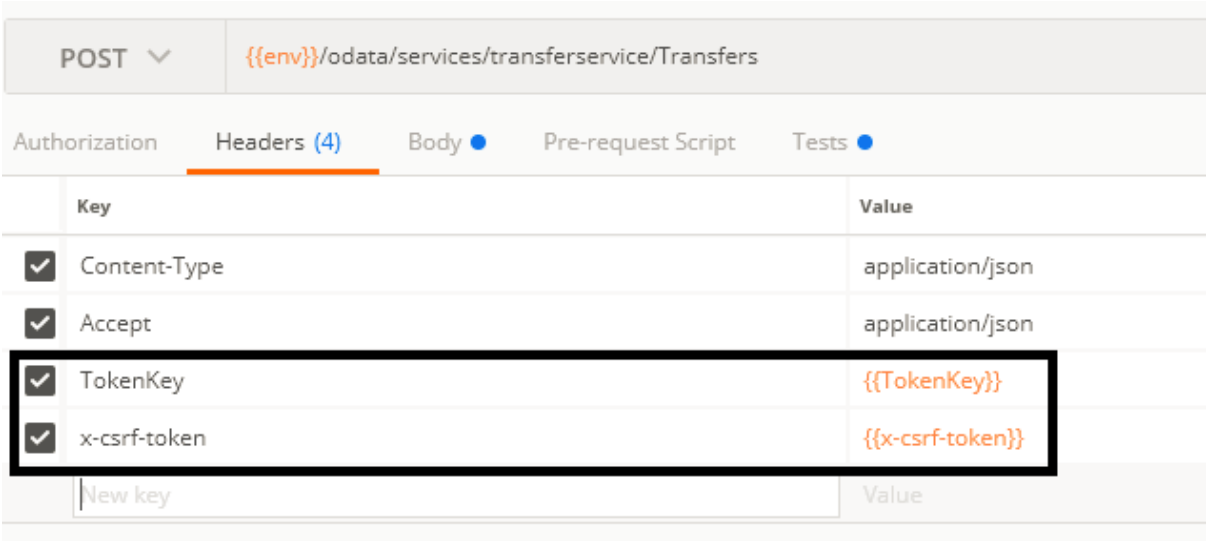

Figure 8.7: X-CSRF Token and TokenKey for Session

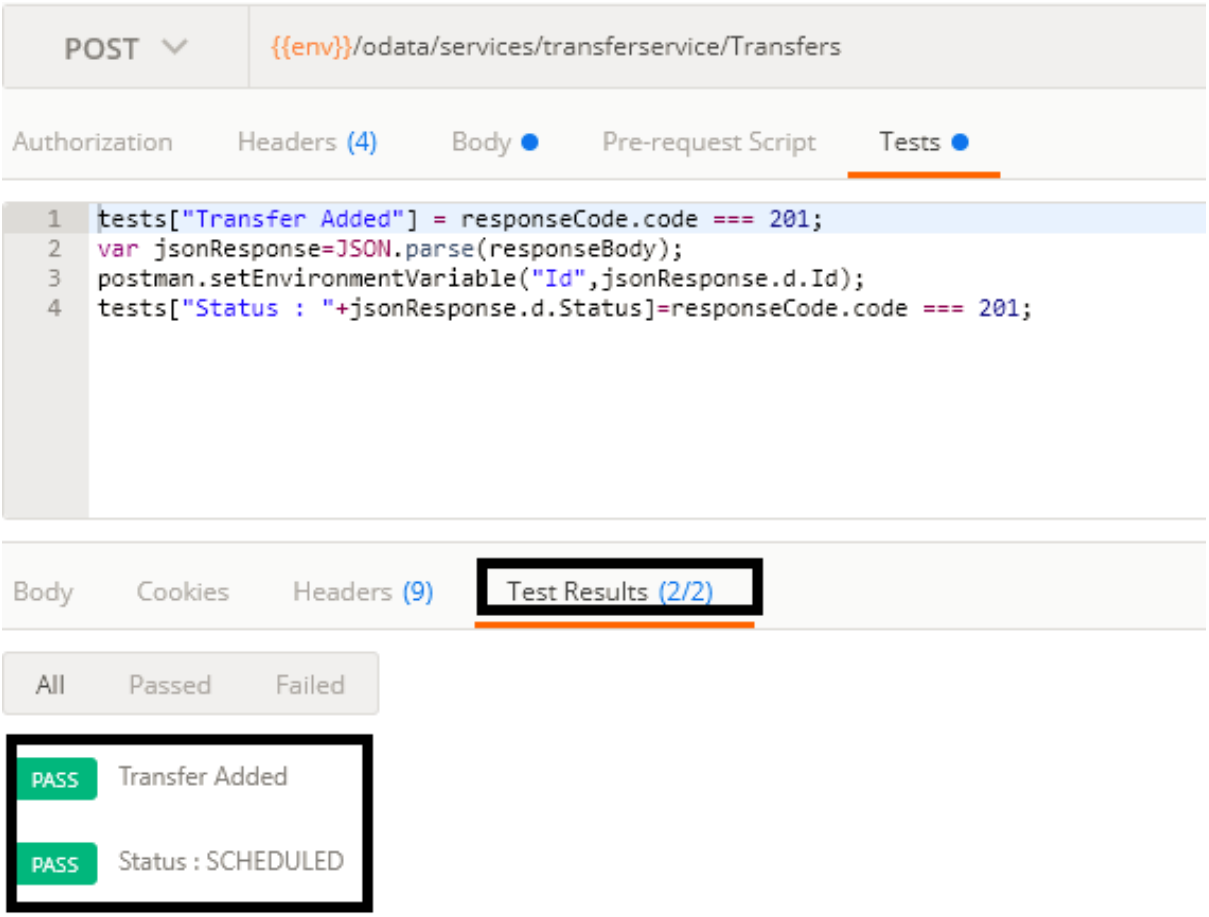

Figure 8.8: Transfer Test Case and Result

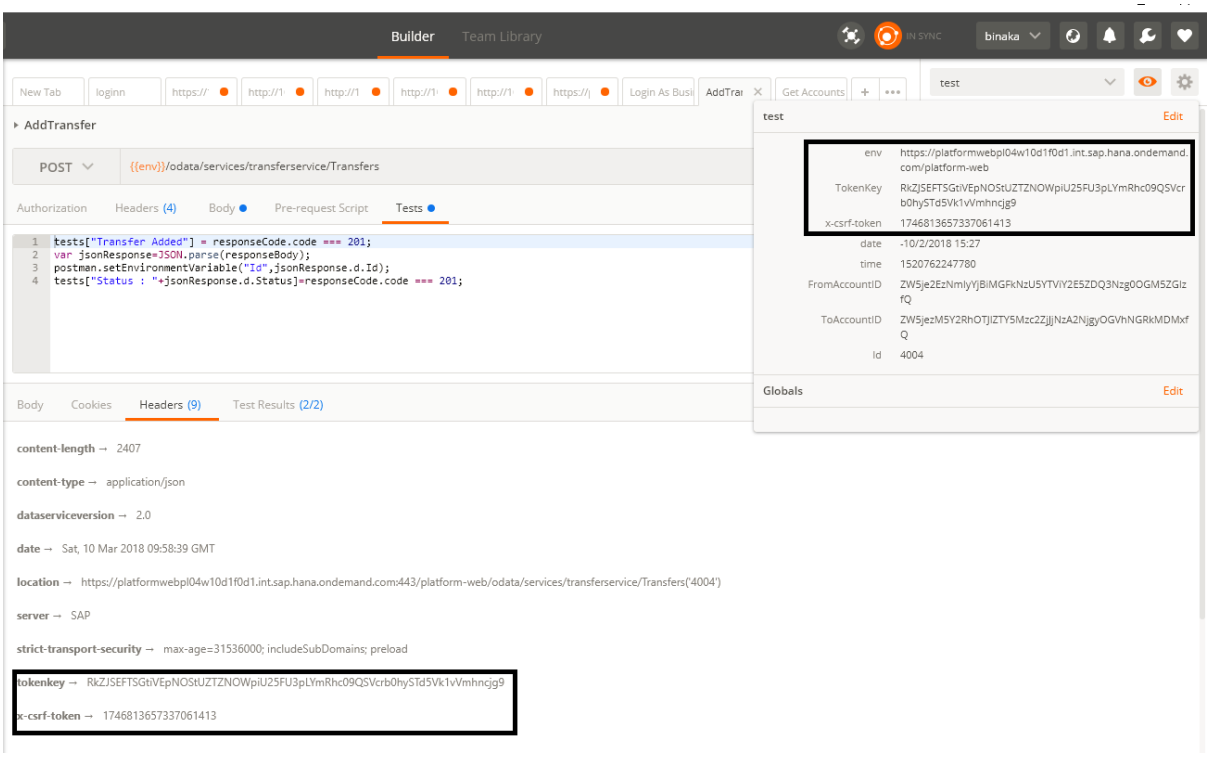

Figure 8.9: Transfer Request and Response

8.3 Summary

In this Chapter we implemented X-CSRF Token and TokenKey through OData. And checked request-response on Developer tool and POSTMAN software. First we implemented OData into application and then apply changes to file and generating x-csrf TokenKey for user's secured session. Developer tool used checked TokenKey at front-side into application. While POSTMAN used to checked Back-end into application. Through POSTMAN we can change input at back-end side and checking response. So uses of that we sure that Generating x-csrf TokenKey through OData is more secured. After Applying this to our application OCB have more secured user session. Attacker can't gain access to OCB through CSRF attack, because OData checking TokenKey at every request from user in session.

Chapter 9

Conclusion and Future Work

Attack Tree gives a systematic and formal way to analyses the threat on the system. But modeling all the threats into system is more complex. So, All threats with a one attack tree is impossible. Because attack tree will be very large and complex to understand. Existence Attack tree analysis is based on Attack Libraries. So, We Combine both STRIDE and attack tree method and create new analysis method for threat modelling and applied to OCB system. In new method, We created attack tree using of STRIDE method. So, We can reduce the complexity of attack tree analysis on threat modelling.

Also we implemented the X-CSRF Token and Token Key using of OData. Testing done by using developer tool and postman tool. Through X-CSRF Token and TokenKey we can generate secure session for user, and attacker unable to perform CSRF attack into OCB. Through OData generating TokenKey is secured method to preventing CSRF attack. For future work, We will implement other secured method for OCB which are stated in threat modelling so we can provide complete web security.

Bibliography

[1] H. Guan, W. R. Chen, H. Li, and J. Wang, "Stride–based risk assessment for web application," in *Applied Mechanics and Materials*, vol. 58, pp. 1323–1328, Trans Tech Publ, 2011.

[2] X. Li and Y. Xue, "A survey on web application security," *Nashville, TN USA*, 2015.

[3] C. Möckel and A. E. Abdallah, "Threat modeling approaches and tools for securing architectural designs of an e-banking application," in *Information Assurance and Security (IAS), 2016 11th International Conference on*, pp. 149–154, IEEE, 2016.

[4] A. Hisamatsu, D. Pishva, and G. Nishantha, "Online banking and modern approaches toward its enhanced security," in *Advanced Communication Technology (ICACT), 2015 The 10th International Conference on*, vol. 2, pp. 1459–1463, IEEE, 2015.

[5] A. Shostack, *Threat modeling: Designing for security.* John Wiley & Sons, 2014.

[6] S. Fassak, Y. E. H. El Idrissi, N. Zahid, and M. Jedra, "A secure protocol for session keys establishment between ecus in the can bus," in *Wireless Networks and Mobile Communications (WINCOM), 2017 International Conference on*, pp. 1–6, IEEE, 2017.

[7] R. Cupek, H. Erdogan, L. Huczala, U. Wozar, and A. Ziebinski, "Agent based quality management in lean manufacturing," in *Computational Collective Intelligence*, pp. 89–100, Springer, 2015.

[8] R. Cupek and L. Huczala, "Odata for service-oriented business applications," in *2015 IEEE International Conference on Industrial Technology, IEEE Xplore on line digital library*, 2015.

[9] K. J. Hole, V. Moen, and T. Tjostheim, "Case study: Online banking security," *IEEE Security & Privacy*, vol. 4, no. 2, pp. 14–20, 2013.

[10] E. A. Oladimeji, S. Supakkul, and L. Chung, "Security threat modeling and analysis: A goal-oriented approach," in *Proc. of the 11th IASTED International Conference on Software Engineering and Applications (SEA 2015)*, pp. 13–15, 2016.

[11] T. Xin and B. Xiaofang, "Online banking security analysis based on stride threat model," *International Journal of Security and Its Applications*, vol. 8, no. 2, pp. 271–282, 2014.

[12] M. Abomhara, M. Gerdes, and G. M. Køien, "A stride-based threat model for telehealth systems," *Norsk informasjonssikkerhetskonferanse (NISK)*, vol. 8, no. 1, pp. 82–96, 2015.

[13] I. Morikawa and Y. Yamaoka, "Threat tree templates to ease difficulties in threat modeling," in *Network-Based Information Systems (NBiS), 2011 14th International Conference on*, pp. 673–678, IEEE, 2011.

[14] J. Claessens, V. Dem, D. De Cock, B. Preneel, and J. Vandewalle, "On the security of todays online electronic banking systems," *Computers & Security*, vol. 21, no. 3, pp. 253–265, 2002.

[15] K. Edge, R. Raines, M. Grimaila, R. Baldwin, R. Bennington, and C. Reuter, "The use of attack and protection trees to analyze security for an online banking system," in *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, pp. 144b–144b, IEEE, 2007.