

# Understanding the Value and Potential of Threat Modeling for Application Security Design - An E-Banking Case Study

Caroline Möckel<sup>1</sup> and Ali E. Abdallah<sup>2</sup>

<sup>1</sup> E-Security Research Centre, London South Bank University,  
103 Borough Road, London, SE1 0AA, UK  
*moeckelc@lsbu.ac.uk*

<sup>2</sup> E-Security Research Centre, London South Bank University,  
103 Borough Road, London, SE1 0AA, UK  
*A.Abdallah@lsbu.ac.uk*

**Abstract:** Software is the most important line of defense for protecting critical information assets such as in e-banking. The continuous increase in sophistication and in volume of cyber security attacks provides compelling reasons for enhancing the security of software applications that control critical assets. There is a broad acceptance that in order to produce dependable and secure applications, developers need to "build security in" throughout the software development lifecycle (SDL). Threat modeling is essential for building security in at all the SDL stages and in particular at the design stage. In the last few years, several innovative approaches to threat modeling have emerged and recently some supporting tools have become available. Using the Microsoft SDL tool as an example, this paper elaborates, illustrates and discusses the threat modeling process and its usefulness to the architectural designs of an e-banking application. This paper also seeks for a critical reflection on different approaches and tools, including the ACE tool and threat trees, accounting for the complexity and difficulty of the process.

**Keywords:** Threat modeling, Security design, Data flow diagrams, Threat modeling tools, Internet banking

## I. Introduction

On the basis of security trends and developments of the last decade, where vulnerabilities and incidents reported have increased significantly and attacks are constantly getting more sophisticated while requiring less intruder knowledge [1], innovative threat evaluation techniques for computer systems and software are needed. On the business side, security objectives in areas such as identity management, financial risk, corporate reputation and business continuity as well as legal and regulatory perspectives [2] need to be addressed adequately by modern assessment methods.

The historically prevalent reliance on network security, provided by general solutions applied to specific applications such as firewalls, would not overcome logic errors, architectural flaws and other system design problems. The failure to produce secure code at the design and development

stage would eventually lead to the exploitation of present vulnerabilities by an attacker [3,4]. In addition, the relative cost and negative effect on security return-on-investment (ROI) to fix these vulnerabilities proves to be highest after release, the National Institute of Standards and Technology estimates it might be 30 times as high as the cost to correct faults at the earlier design stage [5]. In contrast, threat modeling as a concept promises to raise security to a higher level of abstraction. By understanding the threat profile of the system and the related level of mitigation, threat modeling helps to discover vulnerabilities, serves as a basis for secure design and provides a framework for code reviews and penetration testing in the context of the application's security life cycle [3]. Hence, threat modeling as a structured and formal way of presenting and assessing security risks for a specific application has emerged as an independent and comprehensive methodology [7]. At the same time, threat modeling is not an explicitly mathematical science, but offers a high degree of freedom in application, leaving it open also to non-security experts. Its approach can be asset-centric, attacker-centric or software-centric, dependent on the target system examined and the tool or procedure employed. This is also closely related to the range of tools available for software-based threat modeling as well as the ongoing development of the process during the last few years [6].

The largest stake of research and advancement in the area of threat modeling has been provided by Microsoft, making large parts of their own systematic review for secure software design public, for example through a series of books on the issue such as [3] and [7], their threat modeling software packages such as the ACE Threat Analysis and Modeling tool or SDL Threat Modeling tool as well as concepts for threat classification such as described in STRIDE or risk prioritisation such as embodied in the development of the DREAD methodology [8,9]. While Microsoft has published several success stories for the successful deployment of threat modeling, for example the 45% reduction of vulnerabilities one year after the release of Windows Vista in relation to Windows XP [10], the amount of academic literature dedicated to threat modeling is limited. Only few materials have been focussing on the practical

application of threat modeling principles in business environments [11,12,13,14]. Aspects regarding the usefulness of threat modeling for security issues have not been specified in many documents [3,16] and very few authors have decided to base their research on the Microsoft software tools for threat modeling [13,15]. Even fewer papers have attempted a comparison of different threat modeling methods at this point in time [25].

This paper presents, compares and contrasts several approaches to threat modeling and illustrates their applications to the identification, analysis and understanding of threats relevant to the design of an e-banking application. Of central interest is the Microsoft threat modeling tool in context of the SDL (Security Development Lifecycle). Although this tool has been widely used internally at Microsoft, there is a lack of publicly available literature evaluating the tool based on particular case studies. Starting with the initial design of an e-banking application, the paper shows step-by-step how the tool can be used to systematically identify and analyse the impact of relevant threats. As a result of this process, an arguably more secure design is proposed in this paper, which could be validated by the tool.

This paper is organised as follows. Section II provides the reader with the necessary background knowledge about threat modeling, an overview of its foci, concepts and tools. Section III introduces an exemplary case study on e-banking and is followed by the composition of a threat model using the SDL software tool. The threat analysis description includes data flow diagrams, threat identification, related mitigations and assurance of the model. Section V analyses alternative tools and methods, followed by section VI which concludes the paper with a critical discussion of the issue.

## II. Threat Modeling Foci, Concepts and Tools







### A. Threat Modeling Foci

Before even starting to think about the actual threat modeling process with its specific underlying concepts and designated steps, the question about the focus of the threat model needs to be answered. Various aspects can be in the centre of the modeling process - assets to be protected, the attacker's view or the software architecture of the system. The decision which focus to employ depends on its related limitations, advantages or the used tools and methods. Asset-centric approaches address the protection of assets, understanding and managing business risk. Deployment patterns and business objectives of the examined system will most likely be known, assets and access control will be understood. This makes asset-centric approaches ideal for clearly defined line-of-business applications with very specific aims. Microsoft's Threat Analysis & Modeling (TAM) tool developed by their Application Consulting & Engineering (ACE) team is an example for the practical employment of such an approach. In contrast, software-centric approaches are more suited for systems with an unknown deployment pattern and designed to ensure the security of the software's underlying code in the context of rich client/server application development. The Security Development Lifecycle (SDL) Threat Modeling tool, also by Microsoft, is an example for the usage of a software-centric focus. The attacker-centric focus takes the adversary's view to identify risks to the system. This requires to think like an attacker, to understand their motivations and

abilities, which may pose a challenge to inexperienced users. Attacks trees (also called threat trees) can be used to impart this information [2,20,22].

### B. Concepts Underlying Threat Modeling

Most authors agree on the main steps within the threat modeling process, although there are slight variations in terminology, scope and focus due to the progression of threat modeling over the last years. As high-level steps of threat modeling, [3] mentions understanding the adversary's view, characterising the security level of the system and determining the threats. As a pre-requisite and start to the actual threat modeling process, gathering background information, identifying assets and creating an architecture overviews as well as the identification of security objectives [2,14] are mentioned.

DFD Element Name	Characteristics	
	Symbol	Description
External Interactor		Input to the system
Process		Transforms or manipulates data
Multiple Process		Transforms or manipulates data
Data Storage		Location that stores temporary or permanent data
Data Flow		Depicts data flow from data stores, processes or interactors
Boundary		Machine, physical, address space or trust boundary

**Figure 1.** Data Flow Diagram Elements, Symbols and Descriptions [7]

Decomposing the application, mainly by using Data Flow Diagrams (DFD) alternatively Unified Modeling Language (UML), is seen as an important early step of the process. The classification of elements used by DFDs comprises external interactors, simple and multiple processes with subprocesses, data storages and data flows, visually described by a variety of schematic symbols (see fig.1). With their representation of data flows moving through the system, DFDs help to understand the levels of trust apparent in the system as well as the attacker's view of the system [3]. In [17], a detailed analysis of DFDs has been carried out and potential further development directions have been indicated. This is followed by the identification of threats using the STRIDE mnemonic, which categorises threats as follows: Spoofing identity, Tampering with data, Repudiation, Information disclosure, Denial of service, Elevation of privilege [2,7,13,14].

Element Type	Threat Types					
	<i>S</i>	<i>T</i>	<i>R</i>	<i>I</i>	<i>D</i>	<i>E</i>
External Interactor	✘		✘			
Process	✘	✘	✘	✘	✘	✘
Data Storage		✘	✘	✘	✘	
Data Flow		✘		✘	✘	

**Figure 2.** STRIDE-per-element matrix from [8]

By considering threats of these various categories for each single element in the DFD (referred to as STRIDE-per-element in [8]), STRIDE greatly supports the identification of threats within the application. This is also based on the realisation that only certain threat types will

generally apply to certain elements, e.g. all threats behind the letters S-T-R-I-D-E may affect processes, but only spoofing and repudiation apply to external entities, following the STRIDE-per-element matrix chart mentioned, see fig.2. Other methods to identify threats are threat graphs or structured lists with categories such as network, host or application or the type and motivation of attackers [2]. Attacks trees (also called threat trees) are then often used for further analysis and understanding of identified threats, determining whether vulnerabilities of all assets or threat targets in the system, potentially leading to the successful execution of an attack, have been considered [3, 7].

The original threat model process as described in [7] is then followed by a threat rating, symbolised by the 5 letters D-R-E-A-D for Damage potential, Reproducibility, Exploitability, Affected Users, Discoverability and assigned numeric values representing their impact on the overall security risk. The latest methodology however, as used by Microsoft to date, builds on STRIDE and incorporates four main steps, described as diagramming, threat enumeration, mitigation and verification in [6, 8]. Threat modeling is recognised as a "cornerstone" of the Software Development Lifecycle. In the context of the new SDL for Agile development, threat modeling is seen as a SDL requirement for new features and all changes within an Agile sprint.

### C. Software Tools Supporting Threat Modeling

Threat modeling can be conducted without the usage of any software tools or particular frameworks, but due to its broad extent and coverage, a guided process with specified steps and structured resulting reports may be beneficial for most users. While Microsoft has released two different threat modeling tools, there are a range of threat modeling frameworks and connected tools from various origins and backgrounds available. This comprises approaches similar to the described Microsoft framework such as TRIKE, academic methodologies such as OCTAVE from the Carnegie Mellon University's Software Engineering Institute in collaboration with CERT, frameworks based around standards such as the Australian/New Zealand Standard AS/NZS 4360, governmental systems such as the Common Vulnerability Scoring System initiated by the US Department of Homeland Security and various open-source risk management tools [2]. Since all frameworks have varying advantages and limitations, different systems may be better suited for individual organisations and their particular abilities and requirements than others. There is still a lack of assessment criteria for the quality of threat models and while no model can be said to be superior to another, the implementation of any structured threat model process will yield more results than not using a formal security inspection in the application design at all.

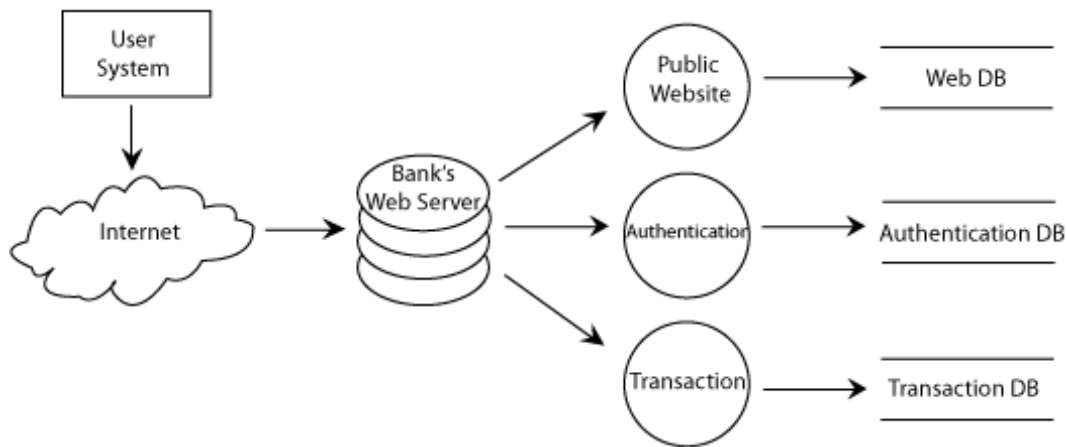
While Microsoft had originally only released one threat modeling tool, the Microsoft Threat Analysis & Modeling (TAM) tool by their Application Consulting & Engineering (ACE) team, this was followed by the public release of the Security Development Lifecycle (SDL) Threat Modeling tool used by product development groups for internal security reviews during application development. These two independently developed tools are both freely available in their versions 3.1 for the SDL tool and 3.0 for the ACE tool, and run on latest Windows systems with Microsoft .NET Framework Version 2.0 and Visio 2007 for the SDL tool. The

reason for this coexistence may be confusing at first, a closer look at both distinct products will however reveal their different approaches to threat modeling, which make each tool suitable for a certain purpose. As discussed in the previous section II A, the TAM tool is seen to take an asset-focused approach suited for line-of-business applications, while the software-centric SDL tool is the official threat modeling tool for software with a broad deployment range developed at Microsoft. Deciding on the right tool to use might be a problem at times, but ultimately depends on the nature of the examined application and the required outcomes for the analysis. For this paper, with its case study on e-banking, it was decided to demonstrate the SDL tool with its DFD and STRIDE elements, while keeping in mind that other methodologies may yield additional or different results to complement the results of the SDL tool analysis.

## III. Online Banking Case Study

Whereas the banking functionality for online banking applications has been derived from its real-world counterparts, designing these system and furthermore securing them has put up an entirely new challenge to banks, service providers, regulating bodies, software developers and information security experts but also customers to follow this change. Fraud figures for online banking are still alarming, with Financial Fraud Action UK reporting a rise of losses by 55% to £39m in the first half of 2009 [23], leaving the banks with an ongoing quest for the best possible security solution for their online banking applications and systems.

A large number of authentication techniques and technologies has evolved over the years, but the distinct methods employed by particular banks vary greatly from simple username and password combinations to hardware tokens with a smart card. Most of these techniques have been introduced as a reaction to fix vulnerabilities discovered rather than pro-actively. Basic authentication techniques protecting users against simple fraud schemes such as social engineering or basic malware such as phishing or keyloggers have been overcome by criminal forces in the last few years. Advanced attacks through complex malware such as spoofing or replay attacks have been the choice of fraudsters to defeat security solutions still employed by banks. This includes one-time-passwords, e.g. the German iTAN solution or basic token solutions as mentioned in [18] or on-screen passwords as used by Lloyds TSB for example. To strengthen the security of their online banking systems, several banks have now introduced offline card readers based on the so-called Chip Authentication Programme (CAP) developed by MasterCard. Barclays UK for example have been largely successful in defending fraud with their CAP adaption, the PINsentry, and there are now a huge range of specialist suppliers offering CAP-based authentication solutions to banks (e.g. VASCO or Reiner SCT, Germany). Besides its positive reception so far, the CAP system has also attracted profound professional and academic criticism, including reverse engineering attempts of the proprietary protocol [18] or penetration testing [19]. In addition to these systems, there are various other solutions available, involving certificates, software or USB smart card readers, showing the fast pace and high level of competition within the market.



**Figure 3.** Schematic and Simplified Overview of the Structure of an Exemplary Online Banking System

Authentication in this case also accounts for the level of risk involved, based on the nature of the protected assets. Most banks will offer a freely accessible public website to unidentified visitors for information and marketing needs (fig.3) without prompting authentication details. In contrast to that, access and usage of online banking functions are restricted and only available to customers registered as users for these facilities by the bank. Normally, banks will employ one authentication method to verify the identity of genuine users and then grant access to the banking portal, but require a second authentication step for transactions of funds as their abuse potentially poses a larger financial risk. The public website, authentication and transaction functions are accessed by contacting the respective web server from the public internet. These processes will also be dependent on their related data bases (see fig.3) and source required data such as web page scripts, authentication data for verification or transaction details and records from there. While real-world online banking applications can only be presented through far more complex schemes and are often dispersed to multiple third-party suppliers other than the bank itself, the demonstrated case study has been simplified to give full credit to the topic of applied threat modeling.

The overall banking institution will have a significantly larger information security environment, including internal LAN networks, related data stores maintained by third parties and other external suppliers potentially creating risks. While a number of assets in this environment may become subject of target-orientated attacks, assets in the focus of online banking security - and therefore this paper - include credentials for authentication purposes, transaction data, personal details and public pages. The basic functional roles in this system can be specified accordingly as visitors, including unregistered users and potential attackers, furthermore genuine users in their position as customers and account holders at the bank and assigned webmasters with additional administration rights. The entire information security environment of the bank will however include a wider range of roles such as auditors, employees with various sets of rights and other IT functions.

It may be expected that attackers will continue targeting these assets in future, either by discovering vulnerabilities of latest security mechanisms or shifting their attacks towards new threat targets. Based on this assumption and the high relevance of secure e-banking to both banking customers and institutions, using a generalised e-banking scenario as presented in the diagram for illustrating the use of software tools for threat modeling seems justified. In this constantly

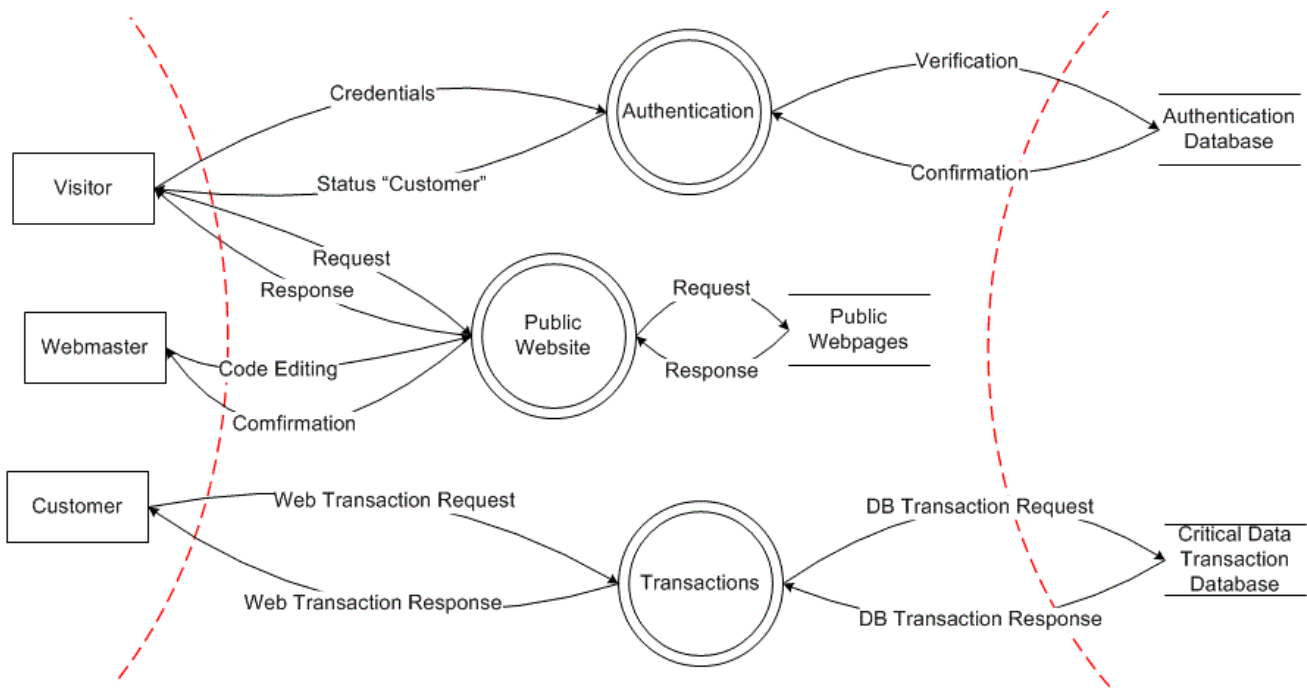
evolving and highly critical security context with its range of potential future threats, threat modeling may prove to be a valuable tool for risk management. While this paper evaluates threat modeling and appropriate tools based on the example of Microsoft's SDL tool and the analysis of online banking security threats, findings from this report may surely be transferred or reproduced for other areas and tools to substantiate the benefits derived from threat modeling.

## IV. Composing the Threat Model

### A. Building Blocks of Threat Modeling in the SDL Tool

To practically demonstrate general functionality, results to be expected by and to show direct outcomes for a real-world example, a threat modeling process for the case of online banking security was carried out with the SDL tool. Four building blocks form the threat modeling process the software is based on: decomposition of the application through DFDs, identification and enumeration of threats using STRIDE-per-element, their respective mitigations and a verification and assurance part. Improving the security of designs, documenting this activity and teaching about security while people work through the threat model have been named as goals of the process in [6] and [8].

In line with the mentioned four building blocks, the SDL tool follows a multi-step procedure, other tools will use a similarly designed approach involving multiple activity layers [16]. The online banking application is firstly fragmented into smaller groups of elements using DFDs. Based on the known structure of the system (see fig.3) and using the provided drawing tools, DFDs can be build in a simple, yet efficient way. Prior experience in creating DFDs is not required, but will be of great advantage, as concepts and elements used in diagrams are not explained in all depth. The elements contained in the DFD will then be related to their respective applicable threat types according to the STRIDE-per-element concept (see fig.2), producing an extensive list of threats to be analysed, described and mitigated. The relation between the provided DFD and the returned threat lists underlines the requirement for a highly sensible DFD representing the system as accurately as possible without being too complex. The verification of the threat model is not subject of the procedure followed by the SDL tool. However, the last two steps, including information on the system's environment and final reports, provide a good overview about employed mitigations, inform various stakeholders and directly prepare the analysis leading to the verification of the model.



**Figure 4.** Data Flow Diagram Edited in SDL Tool, based on Exemplary Online Banking System

*B. Data Flow Diagram of the System*

Like other systems, an online banking application features a variety of components with different levels of internal complexity, trust boundaries, inputs and outputs. Starting on the left of fig.4 (see fig.1 for information on symbols used in DFDs), humans interacting with the online banking system can be viewed as external interactors with no or only limited control over. Data flows describe the way data moves through the system, transferring critical information such as credentials or transaction details, but also simple requests or responses between a public website and its visitors. Processes within an online banking system such as authentication or transaction are likely to be of complex nature and require a logical sequence of simple processes to accomplish their defined task. Simple processes can usually be found at a lower level of the system, which is not subject of fig.4. Online banking systems also contain a number of passive data storages, which hold e.g. authentication, account and transaction details. Furthermore, trust boundaries are significant for online banking, seeing that its general function requires accessing critical data in the form of financial assets, but also information via the public internet by visitors. Using the outlined classification of system components, DFDs can illustrate interactions between elements, show the movement of information through the system and explain the general functionality of the system.

The mentioned external interactors, related to the user system element in fig.3, can be found in the roles of visitors - including attackers - as well as webmasters and customers (see fig.4), all with undefined environments and systems configurations behind their own trust boundary with the public internet on the other side. These external interactors start a number of dataflows through engaging with the central processes of the system. The visitor will request web content from the public website, which will then request data from the public webpages data storage and use the returned data to display the information. The visitor may also send credentials to the authentication process, which will connect to the authentication database to verify the credentials provided by

the user and then confirm the authenticity of the user, which enables the process to assign the “genuine customer status” to the former unspecified visitor. This customer role may request a transaction from the transaction process by providing certain transaction details in text format such as recipient, amount and other details, which is denominated as web transaction request in fig.4. In contrast to that, the transaction process will probably pass on this information in another format suitable for the transaction database and is therefore termed as DB transaction request. The transaction database then responds to the process after the transaction has been approved and processed, for example depending on sufficient account balance. Similar to the authentication database, the transaction database with its critical data content will be behind a trust boundary, separated from the public internet, different to the web server and data storage for the public website. Trust boundaries as plotted in the diagram can help to represent these relations and trust differences appropriately. The last role, the webmaster will be able to edit the code of the public website followed by a confirmation, most likely using a security-critical, web-based content management system.

The DFD may require several revisions to reach a satisfying standard, avoiding an overly complex diagram, but representing the data flows and components accurately. Only a sensible and meaningful diagram will enable an efficient identification of threats using STRIDE-per-element, which is described for the case of online banking security in the following.

*C. Identification and Enumeration of Threats*

Calculated based on the elements included in the DFD, a comprehensive list of approximately 80 potential threats (see excerpt in fig.5) is created by the tool. This threat list enables the analysis of the potential threat impact as well as the assessment of the related mitigation in place within the system. To represent this extensive process, examples for each threat category from this list (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privileges) will be examined in the following.

134	Authentication	MultiProcess	InformationDisclosure
135	Authentication	MultiProcess	DenialOfService
136	Authentication	MultiProcess	ElevationOfPrivilege
37	Public Website	MultiProcess	Spoofing
38	Public Website	MultiProcess	Tampering
39	Public Website	MultiProcess	Repudiation
218	Public Website	MultiProcess	Repudiation
40	Public Website	MultiProcess	InformationDisclosure
41	Public Website	MultiProcess	DenialOfService
42	Public Website	MultiProcess	ElevationOfPrivilege
125	Transactions	MultiProcess	Spoofing
126	Transactions	MultiProcess	Tampering
215	Transactions	MultiProcess	Tampering
127	Transactions	MultiProcess	Repudiation

**Figure 5.** Auto-Generated Threat List Excerpt (Screenshot SDL Tool)

The customer element is selected as an example for the case of spoofing, assuming it would be known to most banking customers because of wide press coverage of phishing cases in the last few years.

Supported by a range of guidance questions provided by the tool (see fig. 6), a number of spoofing threats and their respective mitigations can be defined. Social engineering methods such as phishing to illegally acquire proof of identity or authentication details of the user, e.g. user name, passwords, transaction codes, are a high risk in this case. Malware such as screen- and keyloggers may also be able to retrieve and abuse authentication details. Physical theft of authentication factors, e.g. if the user has written down information, a lost hardware token or the user is forced to disclose information, may be other threats identified. Repudiation is added as another dimension, here, liability in the case of fraud, but also incorrect user input with potential negative financial impact, need to be considered.

As an example for tampering, the authentication database is examined more closely. An attacker may be able to access the authentication database directly by employing an unsupported configuration or executing code on a web server connected to the database and then view, tamper with or store the data contained in this database, e.g. account names and passwords. Repudiation, information disclosure and denial of service threats may also apply to this element, as log files may be altered or deleted, information can be viewed by an unauthorised party and database files can be deleted or the store may be run out of space. This shows the multitude of threats applicable to data storage elements within a system. Because all the data flows connected to the authentication database cross a trust boundary, they should be viewed as particularly security-critical, in contrast to the public webpages database, where all connected data flows lie within one trust boundary.

To expand the example for repudiation as a threat type, threat modeling results for the transaction database containing critical data are outlined in more detail. Specific threats in this category include the alteration of transaction details in favour

of the adversary and the modification of any log file connected to this incident. If there is a general problem with the transaction documentation and log file system, other threats may arise from database administrators intentionally or unintentionally altering data in a harmful manner, evidence problems in case of conflicts with customers may also result from this. Logically, other threat types such as tampering, information disclosure and denial of service threats may also affect this element, as defined in the STRIDE-per-element chart (fig.2) and similar to the case of the authentication database element mentioned before.

The case of information disclosure is particularly interesting for data flows transferring valued assets such as credentials. If the communication channel is not sufficiently protected, attackers may be able to intercept the transferred information and get access to the credentials to use them in a fraudulent manner.

Denial of service attacks will have an impact on many parts of the system, for example the public website. Several critical issues - depending on the underlying infrastructure and coding - such as an connection overload, may make the website unstable and ultimately unavailable to the user, which may affect the reputation of the bank and the overall trust in the security of the bank.

Elevation of privilege threats may affect all processes where permissions are needed to exercise a certain privilege, for example in the case of authentication or transaction processes, where an unregistered user manages to gain access to the banking portal or transaction area and can then view confidential information or conduct fraudulent transactions. This shows that threat types, in this case information disclosure and elevation of privilege, may be related as consequences.

Threat Type Spoofing
<i>Some questions to ask about this threat type</i>
Hint: spoofing is pretending to be something you're not
Are credentials held on the client or server?
Is there a key distribution center?
Are credentials protected in transmission by strong cryptography?
Is there a protocol for updating a credential?
Could an attacker guess credentials (online or offline)?
Could two credentials be mistaken for each other?
Does the app ever support anonymous users or accounts with no passwords?
Does the protocol have a backwards compatibility mode?
Are all credentials random and arbitrary?
Does the protocol always require authentication?

**Figure 6.** Guidance Questions of the SDL Tool

*D. Mitigation of Threats*

The identification and enumeration of threats needs to be complemented by assessing the existence and level of



implementation of relevant mitigations, otherwise the risk posed by certain threats may not be estimated correctly. Looking at the exemplary threats from the previous section, several mitigation measures already put in place by most banks or subject to future implementation can be determined.

Spoofing threats against the customer element can be mitigated by avoiding attack surfaces for social engineering, e.g. through random password generators rather than simple passwords, and offering user awareness programmes and training. Malware attacks can be mitigated by the usage of anti-virus and spyware protection as well as online banking systems resistant to these attacks, e.g. virtual keyboards or randomly selected characters from secret passphrases. Physical loss or theft of credentials can be mitigated by several security recommendations to the user and security design not encouraging violence against the user [14]. User surveillance and password attacks can be mitigated by hiding passwords at the input stage and enforcing the usage of strong passwords. Repudiation at this stage can only be solved by employing a workable policy for handling fraud and other conflicts as well as automatic log files for all user action. While most banks have employed several of these measures in the past, repudiation issues as well as security problems on the customer side prevail.

Tampering with the online banking authentication database can be mitigated by protecting the data storage from direct access through a firewall, private network connection between the web server and the database, sophisticated rights management for viewing, accessing, altering and deleting data as well as physical security measures. Here, verification of input, using parameterised SQL, setting permissions and monitoring access as well as mutual authentication using SSH, PKI or Kerberos over channels protected by standard protocols are indicated as supportive techniques by the SDL tool.

To overcome repudiation threats against databases, but also against other elements of the online banking system, comprehensive logging facilities providing adequate levels of evidence in case of fraud or conflict need to be established. As these threats are often based on prior tampering, reference monitors, access control lists (ACLs) as well as the mentioned mitigations for tampering threats, can help to ensure security. Regulatory requirements as well as a bank's internal audit function will also have impact on preventive measures taken in this context.

Information disclosure of data transferred over data flows can be mitigated by employing standard protocols such as SSL/TLS and extended validation SSL certificates provided by specific certification authorities for correct identification of bank's websites. Most banks in Europe use HTTPS measures for their entire web platform, e.g. German Sparkasse or Santander in Spain, or at least partially for online banking facilities, e.g. Barclays or Lloyds, UK.

Denial of service threats can be mitigated through performance testing for accordant system capacity, input limitation and validation, consistent coding, backup facilities and the existence of an emergency and business continuity plan. This shows the importance of threat modeling for the reduction of operational risk, general risk management, constant high-quality service delivery and the ongoing

learning, improvement and innovation process within the banking corporation.

Protection against elevation of privilege threats can be achieved through sophisticated authentication systems with two-factor authentication, strong authentication factors and two separate steps for identification and transaction authentication. Two-factor authentication is currently employed by a range of banks in Europe ("chipTAN", Sparkasse, Germany; "PINsentry", Barclays, UK), whereas other institutions rely on the strength of their authentication mechanism with one factor ("random characters from passcode with virtual keyboard/pulldown menu", Lloyds, UK; "virtual keyboard", Santander, Spain).

Since the automatic threat generation function of the SDL tool creates a relatively extensive threat list (compare to fig.5), some of these threats may only be theoretically applicable, but do not necessarily pose a severe threat to the system or need to be examined in the threat model. In the SDL tool, threats can only be disregarded after they have been certified by the user as either within a trust boundary, mitigated in another threat model or are considered an accepted risk "per bug bar". Microsoft has transferred the "bug bar" concept from their SDL to threat modeling to be used instead of their earlier DREAD method for ranking threats, to overcome the perceived shortcomings of DREAD such as subjectiveness and impreciseness. As the concept is closely related to the STRIDE model, it assigns a STRIDE threat type, a value for impact on either client or server-side, description of the scope regarding the potential type of attacker, extent and time scale as well as a level of severity ranging from low, moderate, important to critical [24]. In the case study example, threats to the request and response data flows between the visitor and public website may be categorised as an accepted risk, since these are part of the public internet and not concerned with the transfer of confidential data or transaction details.

#### *E. Verification and Assurance of the Model*

On completion of the threat modeling process, the quality, accuracy and efficiency of the model needs to be validated. It needs to be questioned whether all potential threats have been identified correctly and their current level of mitigation has been assessed accurately. Assuring that a threat model represents its real-world counterpart most realistically will ensure that the results derived from the threat modeling process can be translated into seizable, actionable and effective revision plans to improve overall system security. Validated threat model results will also serve as an excellent basis for defending necessary investments for system security in front of superiors in management, technology or security roles.

In [5], Shostack has noted several aspects worth considering for validation at the end of the threat modeling process. DFDs diagrams need to be precise and regularly updated. All threats need to be either mitigated with details on bugs, potential tradeoffs or test plans included or certified as non-mitigated for a valid reason. Specific attention should be paid to data flows crossing a trust boundary and any other element touching these boundaries, here, all STRIDE threats need to be evaluated. While this advice appears very straightforward, the importance of an objective and critical review of all elements, ensuring the quality of an often complex and lengthy process with many participants and a high general degree of freedom,

needs to be stressed at this point. Validation and assurance are also crucial for the long-term development of the system, as changes to the system or its environment may affect its security. Continuous and thorough adaption of the threat model can support maintaining a high level of security and are part of recent development strategies such as Agile. The latest SDL version including threat modeling has given credit to Agile development patterns.

#### *F. Relating to the Environment and Reporting*

Banking institutions may be highly dependent on external entities, outsourcing partners and third parties, for example full-service providers running their online banking systems, operating and computing centres. Especially smaller institutions will not be in the position to absorb large investments for technological innovations and may therefore revert to third party solutions for authentication or transaction services. While large corporate groups with a number of subsidiaries may maintain their own technology functions, their sheer size may cause similar organisational problems. The SDL tool offers the opportunity to note down these external dependencies, external security notes as well as issues for future examination, testing and verification. These implementation assumptions may include the adoption of more sophisticated authentication mechanisms, compatibility of user operating systems and their latest releases, upgrades of the banking software, legal and supervisory regulations and recommendations in future and regular reviews of attack and incident reports. External security notes will for example contain documentation of authentication mechanisms (e.g. Chip Authentication Protocol (CAP) by MasterCard as a basis for many modern security solutions such as Barclays' PINsentry), security policies of the institution and security advice provided by and for 3rd party suppliers. Adding all these dimensions and materials complements the threat model process, but also shows its complexity and resultant difficulty. The report section in the SDL tool provides comprehensive representations of the data input, without proposing any specific further activities. Thus, to benefit from the generated reports and their results, all involved stakeholders and parties need to understand them as fully as possible and translate theoretic outcomes into practical activities to improve security.

## **V. Complementing the Threat Model**

### *A. Attack Trees*

While the STRIDE methodology is perfectly capable of identifying threats, it also needs to be clarified whether the system is prone to become subject to these threats, depending on the existence and status of mitigation of the pre-conditions and requirements for the realisation of these threats. Attack trees with their node and leaf structure can help to specify whether vulnerabilities exist by providing so-called attack paths, which can be seen as the way from the attack root down to each leaf condition [22]. If no mitigation is provided to an attack path, it may be viewed as a vulnerability. The lack of customer awareness on the left side in fig.7 for example is not a vulnerability itself, but can result in actions such as the writing down of credentials, which can then enable physical theft of these and lead to spoofing of customers, giving attackers the possibility to retrieve confidential information or conduct transfers. Following the same consequential sequence, user surveillance may lead to spoofing threats, if

re-usable passwords are in use and an attacker can steal them through shoulder-surfing, if they are not hidden. Schneier has described the full potential of attack trees in [21], including their ability to support the consideration of knowledge about attackers and specific characteristics of attacks such as budgetary, affected users, expected legal consequences or required skill level for launching the attack. This ability can again be illustrated with an example from fig.7: physical theft may only affect individuals, whereas malware attacks over the internet may affect many users, but are not that likely if the attacker lacks specific skills for their execution.

By providing further information on the likeliness of a certain attack in a specified scenario, attack trees can be understood as a structured methodology for analysing system security with a slightly different perspective than the SDL tool. After system architecture, potential threats and attack goals have been analysed with the SDL or any alternative tool, attack trees may be of high value to complement these threat model results.

### *B. The ACE TAM Tool*

While the limited extent of this paper does not allow for an extensive portrait of the Threat Analysis & Modeling (TAM) tool developed by Microsoft's Application Consulting & Engineering (ACE), its potential for profitable threat modeling, also in conjunction with the SDL tool, should be taken into consideration. As mentioned earlier, the TAM tool has its theoretic focus on assets within line-of-business applications rather than taking a software-centric approach like the SDL tool. In practice, the TAM tool does not apply STRIDE but examines threats based on affected roles or components. In strong contrast to the SDL tool, the user is guided through the system decomposition and offered an exemplary task library containing a range of attacks with a selection of suggested mitigations. This assistance may be helpful to novice users or non-experts as it leaves out the creation of DFDs and the definition of specific threats based on STRIDE as well as possible mitigations. Other differences of TAM such as the explicit inclusion of business objectives, access control lists and employed authentication methods may also support other types of users and applications. A positive example of practical employment of the TAM tool in a company is demonstrated in [12], naming advantages such as identifying previously unknown threats and business impacts as well as encouraging risk-based discussions with internal business customers.

In summary, this brief note on the TAM tool indicates that threat models, their results and gained insights may differ based on the underlying tool and method. Considering the complexity of threat modeling, different threat modeling techniques may help to account for all security-related aspects within the system. This is by no means intending to prove that the usage of more than one techniques is useful in any situation, but acknowledging the fact that threat modeling is a multilayered and difficult process aimed at applications and users with a range of backgrounds, various scopes and requirements.



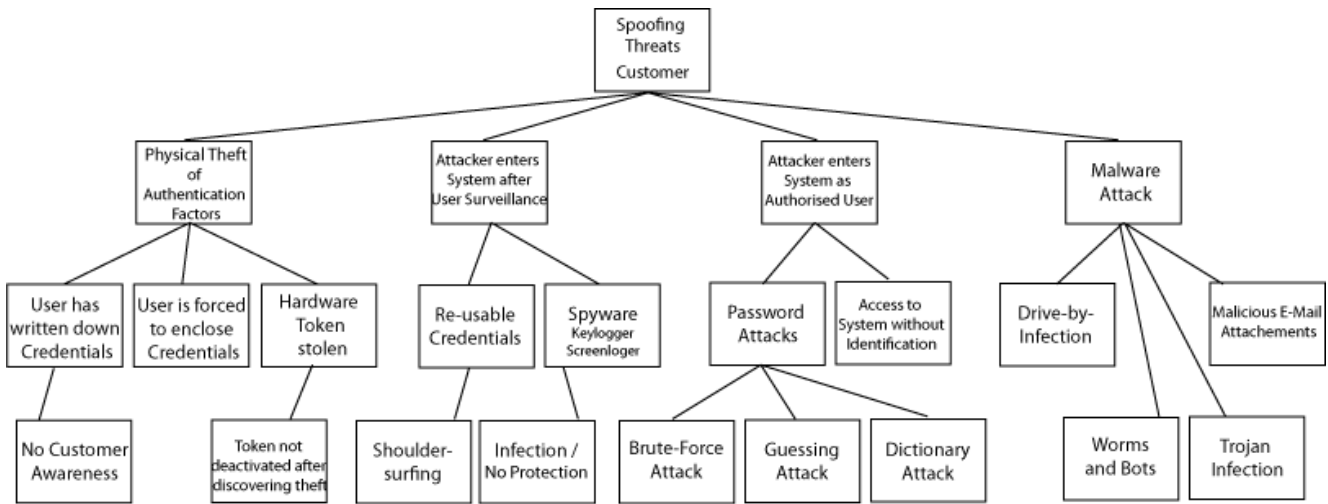


Figure 7. Exemplary Attack Tree for the Case of Spoofing Threats for the External Interactor "Customer"

## VI. Discussion and Conclusion

After the review of basic principles of threat modeling, a clarification of its current status and the practical application of the process to the case study of online banking using the SDL tool, the last section of this paper is dedicated to the discussion of the results from the previous sections. It can be seen as a reflective summary of prior findings, assembling a range of crucial realisations on threat modeling evidenced by the arguments stated earlier.

The general importance of an instance like threat modeling, regardless whether conducted with a software tool or not, is indicated by the rising number of incidents such as security breaches, financial fraud or data losses in recent years, which requires pro-active behaviour rather than delayed reaction to relevant threats. Here, only an abstract, systematic review of system security will enable the early detection of architectural flaws, logic errors and other design problems, driving down the cost and time for their correction, as fixing vulnerabilities proves to be less costly at an early stage of development. Banks and other corporations will require a quantifiable and provable assessment method for system security to integrate security processes into their overall operational processes and to justify their security decisions and investments in an internal or possibly external context. Along the same line of thinking, regulatory issues and legal requirements for banks may also touch on certain aspects of threat modeling.

As this paper aims to evaluate the efficiency of the use of software tools for threat modeling, namely Microsoft's SDL tool, their advantages and limitations are a central point of this discussion. The systematic and rigid nature of the software-assisted threat modeling process becomes apparent in the automatically generated, extensive threat list, which requires the user to consider all potential threats, even unlikely and minor ones. This is crucial for applications with a broad range of deployment patterns, as certain threats may appear negligible to the user, but may become highly critical in certain situations. By working with the threat list based on STRIDE within the SDL tool, users may also learn that a multitude of threats may apply to one element, threats may be based on each other as a consequence and certain threats will affect many parts of the system. Documentation and education purposes are also fulfilled during the exercise of threat

identification in a guided threat modeling process. However, while the analysis of mitigations for threats is also facilitated in the SDL tool, it does not use an overall rating system to assign priorities or levels of endangerment to any threats.

Another important issue of software-assisted threat modeling is the correct perception of its value, ability and limitations – while these tools are a valuable help, they will not create a perfect, customised threat model from scratch, but require sensible input and interpretation at a later stage. Validation of the model and its ability to mirror its real-world exemplar are needed, however the identification of general quality assurance factors for threat modeling seems difficult. This is also related to the expectation of actionable results to be achieved from the threat modeling process. The results and documentation assembled throughout the procedure will only become meaningful and beneficial to the corporation once they are turned into parts of a realistic action plan to improve overall system security. This challenge will not be met by the threat modeling tool, but has to be addressed by individuals with sufficient skills and knowledge, who are able to translate these outcomes into plans and ideas suited to the corporation and its environment.

This leads to the next interesting aspect emphasised by the findings of this paper, the overall complexity of threat modeling. Dependent on influence factors such as the modeled system itself, the nature of the DFD created, the number of participants in the threat modeling process, potential stakeholders and several additional dimensions created by third parties influencing the system security, the process will increasingly become more complex. Threat modeling tools can serve as supportive measures to document and organise all these aspects. However, the exclusive use of one method or tool may not cover all layers of security. As seen in the last sections, other approaches may complement prior results, dependent on their focus, e.g. attack trees can deliver further insight into the likeliness of an attack depending on particular pre-conditions, e.g. the attacker's budget, time or skill constraints.

As mentioned, the threat modeling process, also when supported by tools, offers a high degree of freedom, making it suitable for a range of applications and target groups. This flexibility however requires a relatively open-ended design of

the process, possibly at the cost of novice users. Non-expert users may therefore prefer more guidance, which can be found in the TAM tool, while developers may think in a software-centred way as found in the SDL tool. Catering for the needs of various target groups is a significant difficulty threat modeling has to overcome and current tools have only partially solved, e.g. with a managerial reporting section.

The latest development in the area of threat modeling has certified its inclusion in the agile development lifecycle, which seems reasonable as changes to the system will potentially affect the security of the system. In this context, the usage of threat modeling tools offers the opportunity for reuse and reproducibility of results, making them independent from individuals and available across the organisation.

In the particular context of e-banking, threat modeling may help to keep up with the fast pace of innovation, to identify potential vulnerabilities and avoid their exploitation. Threat modeling can either be directly related to internal security breaches to maintain an updated threat profile of the bank or a common industry effort could be considered. Security measures and authentication methods currently in place can be evaluated and the impact of any planned changes in security policy, system architecture or authentication method can simulated.

Lastly, the question of comparability between various methods and tools for threat modeling in regard to their efficiency and effectiveness remains. Based on the lack of research in the field, the nature and origin of the examined methods, a direct and meaningful comparison may prove difficult and its outcome may not yield results of high value to practitioners and the research community. At this point in time, it seems to be the case, that rather than contradicting each other, threat modeling methods are complementing each other with their different foci, perspectives and scopes, target groups and scenarios. Threat modeling systems of the future will need to strive for adaptiveness, generic and comprehensive underlying frameworks as well as the ability to translate their results into risk-based security decisions.

In summary, overestimating the importance of threat modeling is not possible. While it will naturally happen in an unstructured way in most corporations, a structured approach will offer a range of advantages. In the case of the SDL tool, it will force the user to think about every potential threat, teach them about security and document these efforts, as well as make them reproducible and accessible to many users. Different tools with varying foci may suit different target groups and complement each other's results, accounting for the complexity of the process. This complexity poses a large challenge to threat modeling, even with the use of tools, it remains dependent on the skills of the user, the search for relevant threat rating systems and quality assurance methods has not been overly successful at this point in time and no threat modeling focus or tool will be able to include all security aspects of a system at one time. This does not mean that the current threat modeling methods will not yield interesting results, as indicated through the use of the online banking case study. It is merely the realisation, while threat modeling as a specific concept has shown a tremendous development in its short period of existence, many areas of interest in this field need to be explored and shared by researchers or professionals in the future. This does for

example include formal approaches to the modeling process, concepts for integration into the organisation, development of quality assurance methods for threat models, ideas for a beneficial symbiosis of different methods and tools, but also educational concepts and general coverage including examples to raise awareness for threat modeling.

## References

- [1] W. Stallings. *Cryptography and Network Security – Principles and Practices*, Pearson Education International, Upper Saddle River, NJ, 2006.
- [2] Open Web Application Security Project (OWASP). *Threat Risk Modeling*, available online: [http://www.owasp.org/index.php/Threat\\_Risk\\_Modeling](http://www.owasp.org/index.php/Threat_Risk_Modeling), last accessed: 2010-04-19.
- [3] F. Swiderski and W. Snyder. *Threat Modeling*, Microsoft Press Corp, Redmond, WA, 2004.
- [4] S. Ardi, D. Byers, P.H. Meland, I.A. Tøndel, and N. Shahmehri. "How can the developer benefit from security modeling?". In *Proceedings of the 2nd International Conference on Availability, Reliability and Security (ARES)*, IEEE Press, pp.1017-1025, 2007.
- [5] Microsoft Corporation and iSEC Partners. *Microsoft SDL: Return on Investment*, available online: <http://www.microsoft.de/sdl>, last accessed: 2010-03-25.
- [6] A. Shostack. "Experiences Threat Modeling at Microsoft". *Modeling Security Workshop*, Toulouse, 2008.
- [7] M. Howard and D. LeBlanc. *Writing secure code: practical strategies and proven techniques for building secure applications in a networked world*, 2<sup>nd</sup> ed. Microsoft Press Corp, Redmond, WA, 2002.
- [8] S. Hernan, S. Lambert, T. Ostwald, A. Shostack. "Uncover Security Design Flaws Using The STRIDE Approach", *MSDN Magazine*, November 2006.
- [9] D. LeBlanc. *DREADful*, available online: [http://blogs.msdn.com/david\\_leblanc/archive/2007/08.aspx](http://blogs.msdn.com/david_leblanc/archive/2007/08.aspx), published 2007-08-13, last accessed 2010-04-19.
- [10] Microsoft Security Development Lifecycle Website. *The SDL reduces the Number and Severity of Vulnerabilities*, available online: <http://www.microsoft.com/security/sdl/benefits/measurable.aspx>, last accessed 2010-04-19.
- [11] P. H. Meland and J. Jensen. "Secure Software Design in Practice". In *Proceedings of the 2008 Third International Conference on Availability, Reliability and Security (ARES)*, IEEE Press, pp.1164-1171, 2008.
- [12] J.A. Ingalsbe, L. Kunimatsu, and T. Baeten. "Threat Modeling: Diving Into the Deep End", *IEEE Software*, vol.25 issue 1, pp.28-34, 2008.
- [13] B. Potter. "Microsoft SDL Threat Modeling Tool", *Network Security*, vol. 2009 no.1, pp.15-18, 2009.
- [14] P. Torr. "Demystifying the Threat-Modeling Process," *IEEE Security and Privacy*, vol. 3, no. 5, pp. 66-70, 2005.
- [15] M. Gualtieri, M. Gilpin, C. Wang. *Use Threat Modeling to develop more secure Applications*, Forrester Research, Cambridge, MA, 2009.
- [16] N.A. Malik, M.Y. Javed, and U. Mahmud. "Threat Modeling in Pervasive Computing Paradigm". In *Proceedings of the New Technologies, Mobility and Security (NTMS)*, IEEE Press, pp.1-5, 2008.
- [17] M. Abi-Antoun, D. Wang, and P.Torr. "Checking Threat Modeling Data Flow Diagrams for Implementation Conformance and Security". In *Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp.393-396, 2007.
- [18] B. Borchert. *Online Banking Verfahren*, available online: <http://www2-fs.informatik.uni-tuebingen.de/~borchert/Troja/Online-Banking.shtml>, last accessed 2010-04-19.

- [19] S. Drimer, S. Murdoch, R. Anderson. "Optimised to Fail: Card Readers for Online Banking". In *Proceedings of the Financial Cryptography and Data Security*, Barbados, Springer LNCS, 2009.
- [20] RedTeam Pentesting. *Man-in-the-Middle Attacks against the chipTAN comfort Online Banking System*, available online: <http://www.redteam-pentesting.de>, published 2009-11-23, last accessed 2010-04-19.
- [21] B. Schneier. *Attack Trees*, available online: <http://www.schneier.com/paper-attacktrees-ddj-ft.html>, last accessed 2010-04-19.
- [22] V. Saini, Q. Duan, and V. Paruchuri. "Threat Modeling Using Attack Trees", *Journal of Computing Sciences in Colleges*, vol.23 issue 4, pp.124-131, 2008.
- [23] UK Payments Administration. *Financial Fraud Action UK announces latest Fraud Figures*, available online: <http://www.ukpayments.org.uk/mediacentre/pressreleases/-/page/732/>, published 2009-10-07, last accessed 2010-04-19.
- [24] B. Sullivan. "Add a Security Bug Bar to Microsoft Foundation Server 2010", *MSDN Magazine*, March 2010.
- [25] A.L.Opdahl, G. Sindre. "Experimental comparison of attack trees and misuse cases for security threat identification". *Information and Software Technology*, vol.51, pp.916-932, 2008.

## Author Biographies

**Caroline Möckel** is a second year PhD student at the E-Security Research Centre at London South Bank University. She holds a MSc in international business (awarded in 2009 with distinction) from Fachhochschule Mainz, Germany, and London South Bank University and a BA in multimedia computing (1st class, 2006) from Cork Institute of Technology, Ireland, Oulu Polytechnic, Finland, and Hochschule Darmstadt, Germany. Her research interests lie in the field of e-banking security, usability for security, risk assessment and management, information assurance as well as e-commerce, internet business models and the digital future in Europe.

**Ali E. Abdallah** is a professor of information security, head of the E-Security Research Centre and director of the Information Assurance MSc degree at London South Bank University. He was awarded his MSc and DPhil in computation from Oxford University Computing laboratory and Wolfson College. Prior to his current appointment, he was a lecturer in Computer Science at the University of Reading and a Research Officer at Oxford University. His research interests include software assurance, secure software development, identity management systems, access control and virtual organizations.