Discussion of Information Security in E-Learning

Vom Fachbereich 12 Elektrotechnik und Informatik

der Universität Siegen

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
(Dr.-Ing.)

Genehmigte Dissertation

von

Diplom-Informatiker Christian Josef Eibl

1. Gutachterin: Prof. Dr. Sigrid Schubert
2. Gutachter: Prof. Dr. Andreas Schwill

Tag der Einreichung:	08. Februar 2010
Tag der mündlichen Prüfung:	17. Mai 2010

gedruckt auf alterungsbeständigem holz- und säurefreiem Papier

Abstract

The implied need for security and connected requirements by the use of e-learning systems has only been marginally examined up to now. A problem of respective research is given by the relation of abstract criteria and requirements from educational science with technical realisation possibilities from informatics subdomains. The research project as presented in this thesis focuses on this interdisciplinary field and investigates e-learning from perspectives of both disciplines to develop a security concept that provides a sufficient level of security without negatively influencing the learning process. This thesis, first, deals with the question of which data and functionality for teaching with e-learning systems are reasonable. For this, design criteria for e-learning basing on educational scientific findings are deduced and brought into relation with aspects of information security. Special consideration is put into the learning process and specific requirements of learners in order to prioritise learning in opposition to the pure use of informatics systems. Requirements for an appropriate processing and storing of needed data and functionality are discussed.

Core component of this work is the investigation of threats for e-learning by using the fault tree analysis approach. To conduct this analysis, certain demands resulting from educational science and e-learning research are transferred to technical assets. These assets, thereafter, can be used as root elements of subtrees within the fault tree analysis, i.e., harming such assets is considered to be an undesired event that should be avoided. During the threat analysis process, dependencies on base components of e-learning as web-based service are also considered and discussed, e.g., web server or database system. A central aspect of this analysis is given by mutual dependencies and different expectations of affiliated roles and groups of users in those roles. Exemplary threats, as discovered within the analysis step, are discussed with possible consequences and relevance for e-learning. The entire fault tree is presented in the appendix of this thesis. In addition to the theoretical investigation, case studies of wide-spread and well-known learning management systems are presented that demonstrate different practical shortcomings in the security concepts of these systems. This thesis primarily deals with conceptual problems in the software design, i.e., threats by abusing functionality that was meant to be present in the software. These shortcomings in design imply a significant challenge for mitigation, such that the necessity of an early integration of security deliberations and recommendations into e-learning systems gets emphasised.

Basing on discovered threats for e-learning, respective recommendations are introduced and discussed that can help avoiding the majority of mentioned problems and threats. Knowing that security mechanisms may require user interaction, and therefore, can distract learners from their learning process, a proxy server approach is presented. This proxy server can overtake user interactions following predefined rules. Thus, the user can be relieved despite the implementation of sophisticated security mechanisms. A respective architecture, specification details, and realisation deliberations are discussed together with related problems. This thesis is concluded by a discussion of achieved results and open problems.

Kurzfassung

Der Sicherheitsbedarf und die Sicherheitsanforderungen, die durch die Verwendung von E-Learning-Systemen gestellt werden, sind bisher nur sehr marginal erforscht worden. Problem bei entsprechender Forschung ist die Verbindung von abstrakten Kriterien und Anforderungen aus der Erziehungswissenschaft mit technischen Realisierungsmöglichkeiten wie von Informatik-Teilbereichen geboten. Das hier beschriebene Forschungsprojekt setzt an dieser interdisziplinären Stelle an und untersucht E-Learning ausgehend von beiden Disziplinen, um ein Sicherheitskonzept zu entwickeln, das hinreichend starke Sicherheit bietet, während Anforderungen durch den Lernprozess nicht vernachlässigt werden. Diese Arbeit beschäftigt sich daher zuerst mit der Frage, welche Daten und Funktionen für die Lehre mit E-Learning-Systemen sinnvoll sind. Hierfür werden Designkriterien für E-Learning basierend auf erziehungswissenschaftlichen Erkenntnissen erarbeitet und diese in Verbindung mit Aspekten der Informationssicherheit gebracht. Besonderer Wert wird auf den Lernprozess und die speziellen Anforderungen von Lernenden gelegt, um das Lernen gegenüber der bloßen Verwendung eines Informatiksystems zu priorisieren. Anforderungen an eine entsprechende Verarbeitung und Speicherung benötigter Daten und Funktionen werden diskutiert.

Kernbestandteil dieser Arbeit ist eine Untersuchung bezüglich der Gefahren für E-Learning mit Hilfe der Fehlerbaumanalyse. Um diese Analyse durchzuführen werden Anforderungen, wie sie sich aus der Erziehungswissenschaft und der E-Learning-Forschung ergeben, auf technische Assets abgebildet. Diese Assets dienen im weiteren Verlauf der Analyse als Wurzelknoten für die (Teil-)Fehlerbäume, d.h. als unerwünschte Ereignisse, die vermieden werden sollen. In der Gefahrenanalyse wird berücksichtigt, dass E-Learning als webbasierter Dienst auf weiteren Basiskomponenten wie Web-Server und Datenbank-System aufbaut und folglich Abhängigkeiten von diesen Komponenten existieren. Zentrale Aspekte bei der Analyse sind gegenseitige Abhängigkeiten und unterschiedliche Anforderungen und Wünsche bei beteiligten Rollen bzw. Gruppen von Anwendern in diesen Rollen. Mögliche Gefahren, die sich im E-Learning ergeben und in der Analyse aufgedeckt worden sind, werden exemplarisch diskutiert mit möglichen Auswirkungen und ihrer Relevanz für E-Learning. Der vollständige Fehlerbaum wird im Anhang präsentiert. Zusätzlich zur theoretischen Untersuchung werden Fallstudien mit weitverbreiteten und bekannten Lern-Management-Systemen präsentiert, die verschiedene praktische Unzulänglichkeiten im Sicherheitskonzept dieser Systeme aufzeigen. In dieser Arbeit werden vorrangig konzeptbezogene Probleme behandelt, d.h. Gefahren durch Missbrauch von Funktionen, die im Design der Software so vorgesehen sind. Diese Fehler im Design stellen eine signifikante Herausforderung für die Fehlerbereinigung dar, was die Notwendigkeit einer frühen Integration von Sicherheitsüberlegungen und Empfehlungen in E-Learning-Systeme unterstreicht.

Ausgehend von aufgedeckten Gefahren für E-Learning werden entsprechende Empfehlungen aufgeführt und diskutiert, mit deren Hilfe sich ein Großteil dieser Probleme und Gefahren vermeiden lässt. Mit dem Wissen, dass viele Sicherheitsmechanismen Benutzerinteraktion erforderlich machen, und damit auch Lernende vom Lernen abhalten können, wird zudem ein Proxy-Server vorgestellt, der entsprechende Interaktionen nach vorgefertigten Regeln übernehmen kann, um den Lernenden zu entlasten. Hierfür werden eine entsprechende Architektur, Spezifikationsde-

tails und Realisierungsüberlegungen zusammen mit aufgetretenen Problemen diskutiert. Den Abschluss dieser Arbeit bildet eine Diskussion der erreichten Ziele und offener Probleme.

Vorwort

Die vorliegende Dissertationsschrift ist während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl „Didaktik der Informatik und E-Learning“ der Universiat Siegen entstanden. Eine ganze Reihe von Personen, die ich hier nicht alle namentlich nennen kann, haben während dieser Zeit dazu beigetragen, dass ich die Arbeit erfolgreich abschließen konnte. Besonderer Dank geht an Prof. Dr. Sigrid Schubert, die mir immer wieder Rückmeldungen zu meiner Forschungsarbeit gegeben und das Forschungsprojekt begleitet hat. Außerdem danke ich Prof. Dr. Andreas Schwill für die Übernahme des Koreferats und die hilfreichen Anmerkungen, die zur Präzisierung der Arbeit beigetragen haben.
Für die rege Diskussion fachlicher Fragestellungen möchte ich meinen Kollegen Stefan Freischlad, Peer Stechert und meiner Kollegin Kirstin Schwidrowski danken. Sie waren nicht nur immer wieder bereit, Fragen zu diskutieren, sondern haben darüber hinaus auch Anregungen für weitere Forschungsschritte und -ausrichtungen gegeben.
Ich möchte mich bei allen nationalen und internationalen Kollegen bedanken, die immer wieder den Bedarf der Forschung in diesem Bereich durch Anekdoten und eigene Forschungsfragen unterstrichen haben. Namentlich sei hier Prof. Dr. Basie von Solms, Universität Johannesburg, Südafrika, genannt, der durch seinen Besuch in Siegen im Rahmen eines Kolloquiums maßgeblich zur Definition des Forschungsbedarfs beitrug.
Außerdem danke ich den Studierenden, die mit der Entwicklung des Proxy-Servers „proXsi“ zur transparenten, digitalen Signierung von E-Learning-Inhalten den im Rahmen der Forschung konzipierten und in dieser Arbeit beschriebenen Prototypen eines Sicherheitsagenten realisiert haben.
Ganz besonderer Dank gilt meiner Familie, meiner Frau Gudrun und meinem Sohn Melvin für die ausdauernde Unterstützung und geduldige Toleranz meiner Abwesenheit in den zahlreichen Stunden des Schreibens.

Siegen, im Mai 2010

Christian J. Eibl

* Special thanks to Lori Tilson and family for their great effort in proofreading this dissertation thesis. The given remarks were very helpful.

Contents

List of Figures

List of Tables

List of Abbreviations

ACM	Association of Computing Machinery
AJAX	Asynchroneous Javascript and XML
API	Application Programming Interface
ARP	Address Resolution Protocol
BGP	Border Gateway Protocol
CC	Common Criteria
CERT	Computer Emergency Response Team
CMS	Content Management System
CSCL	Computer-Supported Collaborative Learning
CSS	Cascading Style Sheet
CVSS	Common Vulnerability Scoring System
DMZ	Demilitarized Zone
DNS	Domain Name System
FAQ	Frequently Asked Questions
FMEA	Failure Modes and Effects Analysis
FTA	Fault Tree Analysis
HAZOP	Hazard and Operability Studies
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICT	Information and Communication Technology
ID	Identity
IEC	International Electrotechnical Commission
IEEE	Institute for Electrical and Electronics Engineers
IFIP	International Federation for Information Processing
IMS	Instructional Management Systems
IP	Internet Protocol
ISO	Internation Standard Organization
ITSEC	Information Technology Security Evaluation Criteria
LCMS	Learning Content Management System
LDAP	Leightweight Directory Access Protocol
LIP	Learner Information Package
LMS	Learning Management System
OKI	Open Knowledge Initiative
OSI	Open Service Interconnection
PHP	PHP Hypertext Processor (recursive abbreviation)
PKI	Public Key Infrastructure
QTI	Question and Test Interoperability
RBAC	Role-Based Access Control
SCORM	Sharable Content Object Reference Model
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
URL	Uniform Resource Locator
WCCE	World Conference on Computers in Education
XML	Extensible Markup Language
XSS	Cross-Site Scripting

Chapter 1

Introduction

1.1 Motivation

Today's society is developing towards a knowledge society, i.e., knowledge and the actual degree of information a company or person has gained can be a crucial advantage over competitors. Consequently, the access to basic education in fields relevant for particular tasks and also access to continuing education in this or related fields increase in value for persistence on the market. However, when considering employees with little time available due to their business tasks, the attendance to educational events can get very problematic.

E-learning offers an opportunity for these persons to schedule their own participation in courses and work with learning material in accordance with their business duties. With respect to this target group and the related lack of time, security and reliability of e-learning systems gain in importance. Learners are not willing to spend time on wrong contents or lose time waiting for a service to come up again after system failures. Although e-learning has already been in use for several years and is still promoted as a future trend for learning (not only for the mentioned target group), neither a systematic security analysis of this interdisciplinary topic is available nor a framework to avoid security incidents. Instead, many e-learning systems on the market contain conceptual problems that can easily result in security breaches that would have a bad impact on the institution's reputation if exploited. Recommendation X.800 of the International Telecommunication Union (ITU) illustrates the relevance of security investigation by the following reasons that are relevant to e-learning as well:

1. "society's increasing dependence on computers that are accessed by, or linked by, data communications and which require protection against various threats;
2. the appearance in several countries of 'data protection' legislation which obliges suppliers to demonstrate system integrity and privacy; and
3. the wish of various organizations to use [Open Service Interconnection (OSI)] recommendations, enhanced as needed, for existing and future secure systems." [ITU91, p. 32].

Especially in the context of increasing interconnections, security and reliability of e-learning systems become essential quality factors. Connecting learners over the Internet in this case is part of the goal to reach potential learners all over the world, and therefore, the situation of using publicly available and probably insecure infrastructures must be considered for the analysis. Although information security engineering and underlying technologies like cryptography are almost omnipresent and well-known in informatics, they are mostly insufficiently incorporated or neglected [Sol05]. Security frameworks and recommendations for business processes can be found in numerous variants, but in most cases they are very generic and less applicable for the special domain of e-learning. Thus, it is the motivation of the author to enable e-learning to become an enduring and successful approach by considering security aspects that are specific for this domain and that are related to the primary goal of these systems: support the learning process to the best extent possible.

1.2 Outlining Fundamental Terms for this Discussion

Since no widely accepted definition is available for the most relevant terms in e-learning, the terms “e-learning”, “blended learning”, “learning environment”, and “interaction” will be clarified in the following. In addition, relevant aspects of “information security” with related terms like “threat” or “vulnerability” will be delineated.

Educational Terms

The term “e-learning” in this context will be used with reference to the definition by Kerres [Ker01]. In his opinion, e-learning is a very generic term describing all variants of Internet-based teaching and learning offerings [Ker01, p. 14]. Kerres indeed uses “Internet-based” not only in the often mistakenly used sense of the World Wide Web (WWW), he also includes other protocols and program types like e-mail, news systems, or similar to provide additional means of communication and cooperation. Due to increased desires for personal contact during the learning process (cf. [Lin77, LW91]), the term “blended learning” was introduced as an extension to e-learning. Blended learning describes a current trend of merging presence education with distant learning. However, numerous definition attempts can be found in literature. Stacey and Gerbic analysed such definition attempts and created the following one:

> “In our application of the term blended learning, [information and communication technology (ICT)] may be used to either enhance the dominant mode of face-to-face on-campus interaction and or may provide a blend of synchronous and asynchronous media (that can also include face-to-face classes) to complement a dominant mode of distance education.” [SG06, p. 3].

The inclusion of face-to-face classes, of course, influences exercises as well as communication efficiency. Although the definition by Stacey and Gerbic is very valuable in general, it includes the case of having dominant face-to-face situations. This explicitly will not be considered as the standard situation in this thesis. Thus, in the following, blended learning will be understood as a combination of dominant distance education (e-learning) with a face-to-face part of at least 20% at sensible points in time during a course, e.g., beginning and end of a course as well as prior to a group division for collaborative works.

With this in mind, the term “learning environment” can be specified. Kerres provides a definition as follows:

> “A learning environment consists of different learning offerings and supportive arrangements of personal and (infra-)structural kind. These arrangements should enable different learning experiences and correspond to different learning needs.” [Ker01, p. 316] (translated from German)

In this thesis a learning environment additionally is considered to be Internet-based with regard to the e-learning definition as given above. Furthermore, activities of learners within a learning environment gain in importance especially with respect to interactive multimedia components made possible with modern informatics systems. Thus, the term “interaction” will be regarded more closely. Schulmeister [Sch05] brings “interaction” in e-learning into relation with psychological learning theories, such that he clearly distinguishes interaction from trivial navigation elements. While navigation only means to control which page to be presented and to control the process of an animation or simulation, interaction in his opinion is meant as actually working with a specific learning object, the topic, or the content of a page. Interaction in opposition to navigation, therefore, also includes the manipulation and changing of objects to gain new insights and possibilities to work with and influence these elements.

These educational terms, on the one hand, will be used in the following investigation and discussion, since their specific meaning is crucial for a proper understanding of stated results. On the other hand, the definitions already implicitly exclude certain types of systems and scenarios that cannot be included in this investigation due to the limited space and time available for this research project. Explicit exclusions of systems and limitations in the scope of this research will be given in Section 1.4.3.

Information Security Terms

Information security is related to several terms like "asset", "threat", "vulnerability", and "security services". An overview of the relation of (most of) these terms is given by the International Telecommunication Union (ITU):

> "The term 'security' is used in the sense of minimizing the vulnerabilities of assets and resources. An asset is anything of value. A vulnerability is any weakness that could be exploited to violate a system or the information it contains. A threat is a potential violation of security." [ITU91, p. 32]

Parker [Par81, p. 39] introduced security as a "fuzzy concept", since it is very difficult to handle in a general sense due to differences in every domain. This is supported by several views on security emerging in literature. Different dimensions of security can be distinguished and must be considered for providing proper security recommendations as described in this thesis (cf. Fig. 1.1). Thus, three dimensions will be examined more closely: temporal placement, logical layers, and the impact of security mechanisms. First, the temporal dimension, for example, as focused by McCarthy and Campbell [MC01], describes at which chronological position attack mechanisms are placed:

> "Today's security systems tend to focus on prevention, but that addresses only one-third of the need. [...] Without detection and response components, a prevention-only scheme is destined to fail." [MC01, p. 34f]

Consequently, not only security means and methodologies for preventing incidents, but also for detecting attacks and restoring a system after a successful attack have to be implemented (cf. [Par81, ITU91, Cen02]). In Figure 1.1 this is depicted by a classification of means and methodologies into "prevention", "detection", and "response". Second, the layer dimension represents the proximity to the hardware or network. Recent studies (cf. [HS07, BP07]) explicitly focus on the human factor. Not all security aspects can be solved solely by technical means, i.e., user education and psychological aspects [Wes08] also need to be considered appropriately for implementing security successfully. Policies for formal security implementation and informal security aspects concerning user behaviour (cf. [Dhi07, ÅSS07]) also must be integrated into a holistic security concept. Third, with respect to lists of common protection mechanisms (cf. [ESS06]) it is obvious that not all means are of comparable relevance for security. Hence, this dimension takes care of technically necessary means in the case of "perfect" users, mechanisms of supporting character, e.g., for enforcing security policies or implementing conditional workflows (for non-perfect users), or less relevant mechanisms coping with human laziness and obliviousness.

Some authors (cf. [And01, Eck04]) base their definition of security on the existence and protection of so-called "security services", sometimes also referred to as "protection objectives". A common set of such (disjunctive) security services is given by the so-called CIA model including the services "confidentiality", "integrity", and "availability" (cf. [VK83]). Other services as explicitly discussed in [ITU91, ISO89, And01], e.g., access control or authentication, focus on specific mechanisms implicitly included in the CIA model. The service of non-repudiation also will be

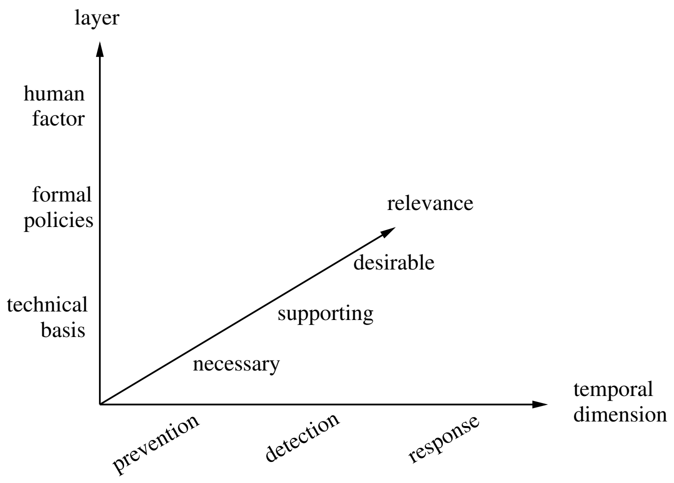


Figure 1.1: Placement of security mechanisms by 3-dimensional vectors

used to emphasise the relevance of being able to trace activities and messages back to their originating user [ITU91, ISO89]. This especially is needed in negotiation processes.

The definition of security as presented at the beginning of this security term section provides a scaffold for threat analyses. First, assets will have to be identified, methodologies will have to be used to discover threats and weaknesses, and all findings will be brought into relation with the dimensions of security and to security services to provide recommendations for minimising vulnerabilities.

1.3 Objectives

Despite the necessity for security considerations in e-learning, there exists no satisfactory analysis or recommendation for deploying secure e-learning systems (cf. Chap. 2). Note that in this thesis the term "e-learning system" is used as the most generic form of a specific software composed of tools and specialised modules with the objective of enabling and supporting e-learning. This term does not state any requirements concerning the architecture, programming language, implemented features, or security concepts.

With respect to the lack of a proper analyses, the following tasks are not sufficiently conducted up to now and represent central objectives of this thesis:

Identification of e-learning criteria
E-learning is an interdisciplinary field, on the one hand, including educational science with learning psychology and didactics, and, on the other hand, informatics with information security and software engineering. In order to achieve a proper notion of security in e-learning, both disciplines must be considered and analysed concerning their impact on security in e-learning. Thus, common criteria should be extracted and discipline specific issues need to be examined with respect to mutually influencing research aspects in another discipline. Due to the vast amount of specialised topics covered by these disciplines, research must be limited accordingly to achieve a manageable subset. Note that criteria of both disciplines might interfere with each other, such that in subsequent steps compromises might be necessary.

Threat analysis and demonstration of case studies
Criteria and dependencies should be considered as starting points for a theoretical threat analysis to uncover possible problems and threat situations for e-learning. For this analysis process, a systematic methodology must be applied to guarantee reasonable outcomes. Discovered threats and possible attacks should be easily transferable to one's own infrastructure. (Theoretical)

analysis results must be approved and evaluated in practical systems using case studies. Existing and widespread e-learning systems have to be considered for this evaluation. Their concepts should be brought into relation with formerly gathered threats. Note that the threat analysis represents the central task in this thesis.

Development of recommendations
Information security and educational science bring in two very different views on e-learning systems. Topics from educational science are quite far from being technically manageable and usable for technical deployment, but they build a certain base of necessities to be fulfilled in order to offer acceptable e-learning systems. On the other hand, models and methods from information security are strongly technically oriented and might restrict the usability of the system only for the sake of security, i.e., they lack contextualisation for e-learning. Hence, an important task in this research project is a transfer of abstract requirements to technically manageable aspects for finding security mechanisms and measures that can be adapted appropriately. This process aims at recommendations for secure e-learning systems that can easily be applied to practical situations and that are sufficiently taking care of the learning process.

1.4 Achieving these Goals

1.4.1 Approach

For investigating such an interdisciplinary topic, the research methodology as sketched in Figure 1.2 will be used. This methodology is structured according to three different aspects:

1. related disciplines,
2. theory vs. practice,
3. phases and milestones.

First, regarding e-learning, we are confronted with two rather contrary disciplines. The “learning” implies pedagogical and related psychological findings to be considered, i.e., educational science, while the “e” is related to informatics systems which in particular for this thesis will be considered from an information security perspective. Since an analysis of this interdisciplinary field needs findings and research results of both disciplines, the research methodology considers these disciplines as columns beside the main column in the centre representing the actual research project with its milestones. From the left column of educational science implications for functionality are extracted. Learning theories and didactic principles (cf. [Dew38, Vyg78, LW91]), expectations/objections according to target group characteristics (cf. [BM93, KHS05]), learning process designs (cf. [Rot63, RS75, Lin77, Stö78]), and theories of learning and understanding (cf. [Dew38, Vyg78]) will be considered for the investigation and be brought into relation with findings during this research project. The discipline of information security contributes to the investigation by classifications of security services (cf. [VK83, ISO89, ITU91]), security models (cf. [Dhi07, ÅSS07]), and analysis approaches for threat discovery (cf. [VGRH81, And01, Eck04, DIN06]). For the elaboration of recommendations for secure e-learning systems, well-known means for hardening applications (cf. [Ris05]), implementing security policy models (cf. [BL73, LB73, Bel74, BL76, And01]), applying cryptography (cf. [Sch96]), implementing countermeasures (cf. [ITU91, Eri03]), and using digital signatures for prototypical helpers (cf. [Sch96]) are used.

As second partition the central column is divided into theoretical and practical parts. Theoretically uncovered requirements and solutions will be transferred to practice and tested accordingly. Since not every theoretical aspect can be transferred in the same simplicity, the assignment of topics in this column takes place with respect to this fact in more or less clear side decisions. This

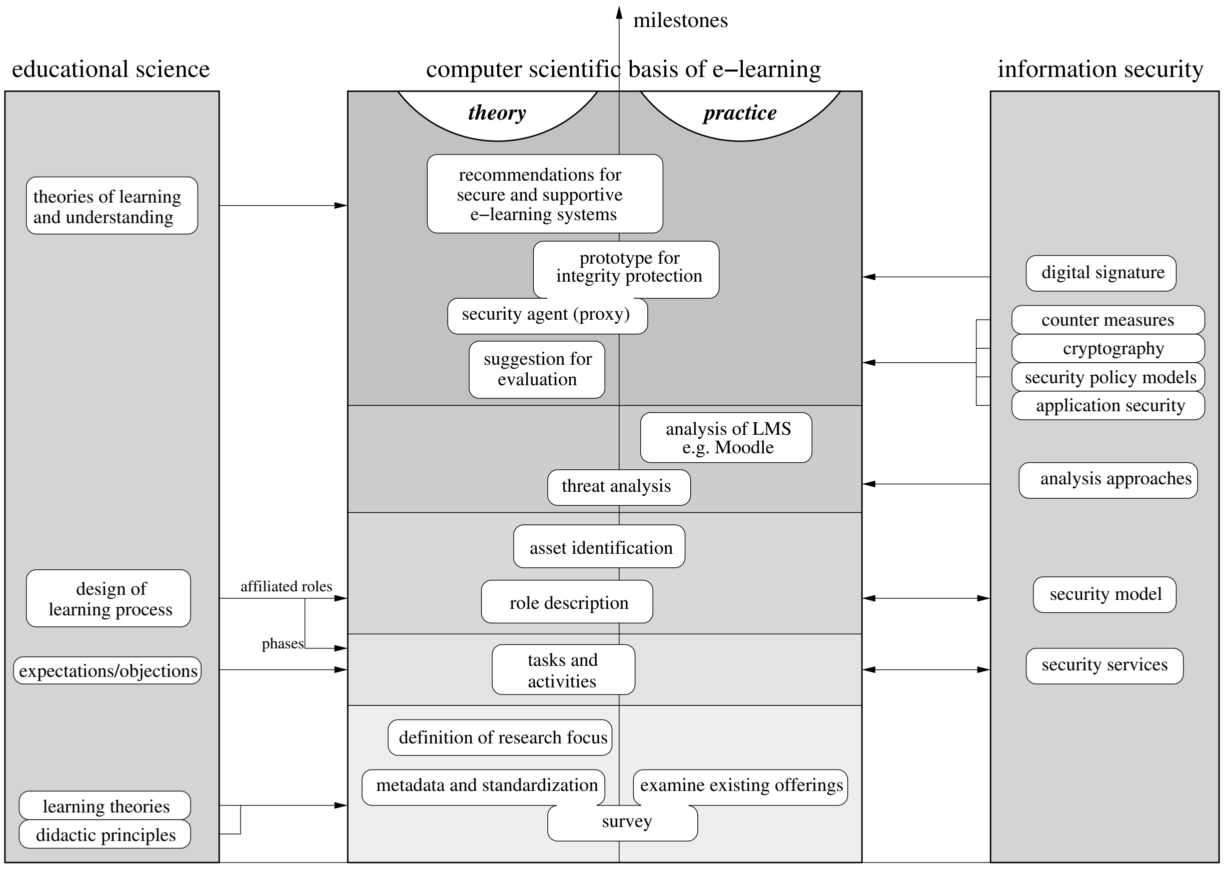


Figure 1.2: Research methodology with related disciplines

subdivision supports the current situation of numerous e-learning systems already in use and the necessity for theoretical analysis and conceptualisation of security concerns. Theoretically uncovered problems can be verified in practice and existing systems can be helpful in completing theoretical deliberations.

Third, this research can be regarded in single phases and milestones. The initial phase, i.e., bottom-most slice in Figure 1.2, is understood as Phase 0 and will not be considered for numbering the steps belonging to the actual analysis process of this thesis. Phase 0 consists of a first survey of the current situation and state of the art in e-learning. Practical projects as well as technical implementations of e-learning systems were examined. Theoretical concepts, technical usage of standards for metadata and content packaging, administrative aspects, and literature related to information security in e-learning were investigated. Research objectives as resulting from this initial survey were already discussed in Section 1.3. For the subsequent phases, an additional alternative figure will be used to introduce and illustrate the current step in each chapter of this thesis with direct relations to neighbouring analysis steps (cf. Fig. 1.3). These steps represent

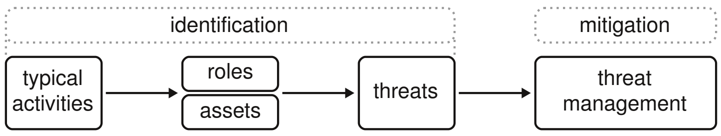


Figure 1.3: Threat analysis process

the analysis as a whole. Starting from e-learning specific activities and data, stepwise more and more threats and problematic concepts are to be uncovered. The respective structure will be loosely based on auditing and evaluation manuals like [MK07, Sei06, SH06]. Thus, the following questions will have to be answered in corresponding phases:

Phase 1: “What functional parts belong to an e-learning system?”
The foremost aim in this phase is to identify typical functionality and related requirements of e-learning systems. Connected to this, pedagogical models concerning the way of learning efficiently as well as didactically relevant elements for technical systems will be included. For the creation of such functionality and activity listings (cf. [Sta02, BDRW03, ES04, KSC05, HLT05]), practical systems will be examined as well as additional tasks will be designed in theory. The theoretical part is supported by expectations and objections of learners according to psychological and pedagogical models with respect to theories concerning the learning process. The organisational phases of the learning process in this thesis play a major role, since every phase of this process should be transferable to the system (cf. Sec. 1.4.3).

Phase 2: “Who is involved and what has to be protected (from whom)?”
Concerning non-technical requirements to be fulfilled for successful e-learning systems, a transfer to technically manageable assets is required for a technical threat analysis. For this, besides the pure availability of functions and connected requirements, a consideration of differences among various e-learning types and corresponding sets of affiliated roles is needed. With respect to these roles, activities can be assigned to specific persons and related objections and expectations be included for extracting assets to be protected in an e-learning system. The direct inclusion of role perspectives is aspired for adequately considering interest conflicts between different roles in the system and respective demands for compromises in a practical implementation. Assets as extracted in this phase will be directly used for the subsequent threat analysis step, and therefore, this phase provides a significant preparation step.

Phase 3: “What can harm introduced assets?”
Having clarified the “what to do”, “who will do it”, and “what data are affected” we can start with a systematic threat analysis process in this phase. Due to an emphasis on personal, and therefore, organisational aspects in learning, this threat analysis must take care of personal requirements and human behaviour, too, besides a technical analysis investigating network and hardware layer security problems and mechanisms. As a starting point for uncovering possible threats, existing evaluation catalogues and lists of common weaknesses in (web) applications will be applied (cf. [Swa01, SGF02, Gro03, BSI05, SH06, Her06, MK07, MBPC09, MIT10]). Note that these catalogues and criteria listings are meant to provide a very generic approach to do security audits on arbitrary systems. They are in no way specific to e-learning, such that further refinement and extension is needed to take care of e-learning characteristics. Personal experiences of the author as administrator of several heterogeneous company networks will be used to complete the threat situations and bring conceptual weaknesses into logical relation with active attacks. In addition to a theoretical investigation of threats, practically used e-learning systems will be included in this analysis as case studies (cf. Chap. 5) to provide lively examples on how discovered conceptual weaknesses can result in serious security problems. This phase is intended to result in an illustrative representation of possible threats.

Phase 4: “How to react to identified threats?”
Usually not all threats can be prevented with comparable effort and by the same means. In the final phase (cf. Chap. 6) a set of recommendations for a secure e-learning system will be presented with respect to discovered threats. Demands from educational science and informatics will be respected adequately. Limitations concerning practical realisations will be outlined. In addition to the conceptual recommendations, a prototypical implementation of a security agent for an automated processing of security relevant tasks will be presented. This agent can overtake critical activities from learners and reduce their amount of non-learning related activities respectively.

1.4.2 Evaluation Criteria

The research objectives as activities to be conducted in this research project were stated in Section 1.3 due to the lack of an appropriate investigation of information security for e-learning systems up to now. In addition, for the sake of quality assurance, final results of this dissertation thesis will have to be evaluated. For this, criteria will be introduced in the following and be discussed in comparison to achieved results in Chapter 7. Note that evaluation criteria will not be directly connected to the research objectives because of implicitly contained dependencies. This means, the objective of criteria identification is implicitly included as basis of the threat analysis, and therefore, for evaluating results this will be considered accordingly without separately evaluating these criteria. Furthermore, case studies are considered as part of the threat analysis and provide a means for refining analysis results, such that this also will not be considered separately. Hence, the main focus of evaluation is put on the analysis and the creation of recommendations for secure e-learning systems:

1. *Reasonable conduction of threat analysis with recognising e-learning specialities*

   An objective of this thesis is to analyse the security situation of e-learning systems and to uncover related threats accordingly. For this, a systematic methodology is desired, and therefore, the application of an appropriate methodology should be verified and discussed. To meet special requirements as inherent in the considered application field of e-learning, the actual analysis process has to include respective demands and specific aspects of e-learning to be in contrast to generic security research, which only in parts could solve the stated problem of describing a highly secure e-learning system.

2. *Complete coverage of revealed threats by recommendations*

   With respect to the final recommendations for a secure e-learning system, discovered threats should have been considered and handled by the integration of counter-measures. This criterion, therefore, aims at the consideration of the actual coverage of these threats in comparison to still unsolved threats. In best case, obviously, all threats can be managed by appropriate counter-measures indicated by recommendations.

3. *Reasonable compromise between effort and security gain of recommendations*

   Security mechanisms usually decrease available functionality and introduce additional interaction demands that promptly could distract learners from their learning process. Thus, it is necessary to balance, on the one hand, the effort of implementing and using counter-measures with all possible negative effects on learning and, on the other hand, possible losses in case of security incidents. Consequently, a reasonable compromise is necessary. The success of the balancing process will be discussed according to this criterion in the conclusion of this thesis.

1.4.3 Limitations

Since the terms "e-learning system" and "information security" describe vast areas of possible research, this has to be limited to a manageable amount for this research project. The term e-learning system as stated above does not restrain the applied technologies in any way. However, in this thesis we focus on the most commonly used form of web applications using the protocol HTTP (hypertext transfer protocol) for the data transfer. A limitation of regarded technological aspects is sensible, since different techniques provide different features and also introduce different security problems. Examples for such technologies are Java applications communicating over the Internet using RMI (remote method invocation), applications using CORBA (common object request broker architecture), or web-based services, e.g., basing on SOAP (simple object access protocol) or REST (representational state transfer).

In addition to the actual technological implementation, the functionality and coverage of the learning process provides a second criterion of limitation for this research project. E-learning systems as investigated in this thesis aim at supporting the learning process in all its phases, such that all tasks from bringing learners in contact with learning matter for the first time to actual practising of already known contents and doing (self-)assessments will be covered. Simple learning software supporting only single elements of the whole learning process will be considered where appropriate, but are not in the primary focus of this research. This statement includes specialised software products like assessment systems. Although assessment systems are of high interest concerning security, they are only used for a very small period of time compared to the learning process as a whole. Since usually only the last assessment for doing an exam and being issued a certificate actually has to deal with increased security demands, and therefore, must be incontestable, the former progress in learning and working with learning contents is not regarded. However, assessment systems were exhaustively investigated by Graf [Gra03] and other colleagues at the author's university, e.g., Wismüller and Hoffmann [HQW08, HW08, HWB09].

A third limitation has to be drawn concerning the term information security. Information security consists of several subcategories that are not of equal relevance in this thesis. Security in general also considers technical security, physical security, personal security (safety), or security of sensitive data. With technical security we mean that computers are built appropriately without technical failures and that the infrastructure provides sufficient resources to enable proper working processes. Physical security means that no unauthorised person should be able to physically access computers, printouts of classified documents, or storage media with sensitive data. Touching the area of safety, personal security regards means and mechanisms to be deployed for protecting the users rather than an informatics system. However, in this thesis the term information security is strongly focusing on the security of data within e-learning systems. This primarily means organisational aspects, but will include technical security aspects with respect to availability and reliable data transfer.

Related to information security is the distinction between risk and threat analysis. In the definition of Courtney [Cou77], which is supported in similar words by Schneier [Sch03a], risk is defined as the product of probability of an incident and the loss attributed to this event. Nevertheless, quantitative analyses are considerably more difficult than simple threat disclosure analyses due to the demand for estimations of an incident probability – which do not necessarily provide additional support for creating a security concept. The Common Vulnerability Scoring System[1] (CVSS) outlines this estimation problem by describing incident probability as combination of three partial estimations. First, the general risk of a vulnerability to allow security incidents must be rated by experts by providing comparable values between several security weaknesses of a very different kind. Second, the temporal aspects concerning topicality and the diffusion rate of knowledge about exploitation of the corresponding weakness have to be included. Third, the actual infrastructure has to be taken into account. Practical systems and infrastructures have their own very specific goals and related configurations, such that too general estimations will not be accurate for these particular scenarios. Consequently, security experts from one's own institution are necessary for estimating values accurately. Hence, the fact of involving different groups of experts, which most probably will result in different subjective estimations, and the enormous number of influencing factors (among others temporal aspects) result in problems concerning accuracy of probability estimations and durability of research results. Therefore, in this thesis the term of *threat* analysis without considering probabilities in any way will be preferred, if not mentioned otherwise, over the more critical term of *risk* analysis.

[1]`http://www.first.org/cvss/` [02-02-2010]

1.5 Principal Contributions

The remaining chapters and their principal contributions are as follows.

Chapter 2: Literature Survey
This chapter provides a literature survey focusing on interdisciplinary conference contributions and monographs dealing with e-learning and computer-supported collaboration. As a result of this chapter, a collection of activities will be presented that implies functionality and respective dependencies in supportive e-learning systems from an educational science perspective. This collection will be used in subsequent chapters as basis for extracting technical aspects to be analysed in more detail.

Chapter 3: Preparation of the Threat Analysis
Requirements and activities, as given by educational science, need to be transferred to technically manageable aspects. For this, first, different types of e-learning systems will be discussed with respect to various sets of affiliated roles and functionality to be present. Second, based on formerly introduced tasks, activities, and roles, assets will be extracted. Such assets mostly correspond to data to be protected by an e-learning system. Thus, the result of this chapter will be a set of assets to be considered in more detail during the actual threat analysis process.

Chapter 4: Analysis of Threats in E-Learning Systems
On top of assets given by the former chapter, conceptual and event-based threats will be analysed using systematic threat analysis methodologies. Specific threats for e-learning directly connected to given assets will be introduced, as well as more comprehensive and asset-spanning conceptual and technical threats will be discussed concerning relations to e-learning systems. The result of this chapter will be a visualisation of related threats concerning e-learning systems in general. This analysis will be conducted for most generic systems not specifically making a distinction of given types of e-learning, i.e., the union of functionality of all these types is investigated.

Chapter 5: Case Studies
Theoretical analysis results will be transferred and compared to practically implemented e-learning systems. Considered e-learning systems will be examined concerning their current security concepts and the historical development of these concepts to illustrate different views on security in e-learning. Selected threats as discovered in the theoretical analysis phase will be verified in these systems to emphasise the relevance for practical situations.

Chapter 6: Recommendations
With respect to discovered threats and problems in this chapter, recommendations will be given for implementing an e-learning system with an appropriate security level, i.e., sufficiently secure but still supporting learning without exhaustive distraction by security means. Furthermore, a proxy server acting as a security agent that overtakes security critical activities from learners without distracting them from the actual learning process will be presented.

Chapter 7: Conclusion and Future Work
Finally, results will be summarised and evaluated according to stated criteria. Possible avenues of future research will be discussed.

Chapter 2

Literature Survey

2.1 Overview

The literature survey to be conducted in this chapter represents the first phase of the threat analysis process (cf. Fig. 2.1). This phase aims at examining related work that already covers or is in contact with single areas of this research project. In addition, it aims at identifying and describing e-learning specific activities and requirements to be considered for subsequent phases.

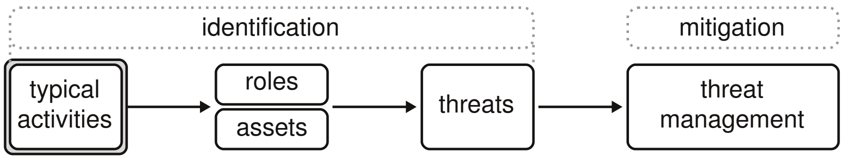


Figure 2.1: Current position in analysis process – step 1

Since e-learning is an interdisciplinary research topic, the search space must be limited to reduce the set of literature to a manageable amount of related work. This will be done in Section 2.2. Closely related work, i.e., literature specifically dealing with information security research in e-learning, will be considered in Section 2.3. However, since only few of such works exist at the moment, the majority of this chapter will focus on more general activities and requirements for e-learning, partially neglecting security for the moment. Section 2.4 will outline the structure and nomenclatures to be used for this approach in subsequent sections. Examining the application field of e-learning in a general manner is meant to practically define the term "e-learning system" to a certain extent, such that further analysis steps can appropriately take care of respective specialities. Thus, Sections 2.5 and 2.6 investigate available literature with respect to implications from educational science and reasonable findings for security engineering. Finally, Sections 2.7 and 2.8 conclude this chapter and summarise discovered limitations and consequences to be considered in further investigation steps.

2.2 Search Space

Due to the vast amount of articles and monographs in educational science, informatics, and, as their interdisciplinary combination, in e-learning, some limitations in the search space for this literature survey are reasonable. This will be done in the following according to two aspects: period of time and research focus.

This research project started in the end of 2005, and therefore, this will be treated as upper boundary for this survey due to its timely limit for an initial investigation of the current situation to that time. With respect to the rapid development in informatics and particularly in the field of e-learning with its multimedia elements and development of web-based services going back more than five years appears less sensible. Publications before 2001 will most probably be outdated or referenced and elaborated by newer articles. This results in a time window of 2001–2005.

Since we strictly investigate the interdisciplinary aspects of e-learning, literature, for example, concerning detailed technical possibilities without sufficiently taking care of domain specific specialities will be neglected. Hamid already pointed out that the "emphasis on e-learning in the past has been on the 'e' [...]" and that there "is a need to shift the emphasis [...] to the learning" [Ham02, p. 313]. Learning specific aspects must be understood, and, related to this, the actual role of the "e" in e-learning must be considered appropriately. To cope with this demand, the limitations in research topics and investigated aspects are ordered by a three steps approach. First, monographs of educational scientists like Kerres and Schulmeister published during the considered period of time will be taken into account. Especially the participation of Schulmeister in the EVA:LERN project (cf. [Sch03b]) that had the aim of evaluating e-learning systems from a didactic perspective motivates this first step. Second, national and international conferences directly addressing e-learning and computer-supported collaborative/cooperative learning (CSCL) will be considered. Concerning related scientific conferences, first of all the World Computer Conference in Education (WCCE) of the International Federation of Information Processing (IFIP) will be regarded. This conference was held in 2001 and 2005 during the considered period of time. Thus, it provides a development overview during this period. The international conference of computer-supported collaborative learning was held in 2001 (as euro-CSCL), 2002, and 2005 during the considered period. Furthermore, the most important German e-learning conference, DeLFI ("Die e-Learning Fachtagung Informatik"), was established and held annually since 2003. In a third step, the survey will be expanded to digital libraries, such that appropriate articles can be integrated into the discussion if arguments are not already confirmed by references of former steps. Considered libraries in first place are the well respected digital libraries of the ACM (Association of Computing Machinery) and Springer. In the ACM library published articles are tagged, such that search can be refined by using keywords and category descriptors. Categories in the ACM library particularly relevant for this research are K.3 ("Computer and Education") and its subgroup K.3.1 ("Computer uses in Education"). As second library Springer as the official publisher[1] of the IFIP will be included. This leads to results and proceedings as published by all technical committees of IFIP, e.g., TC3 ("ICT and Education") or TC11 ("Security and Protection in Information Processing Systems"), as well as covered working groups, e.g., WG 3.6 ("Distance Education").

2.3 (Closely) Related Work

Although in the outlined search space several articles and monographs can be found that address single aspects relevant for this research, there is only little choice in explicit security in e-learning literature. Among others, Eckert [Eck03], Kajava [Kaj03], El-Khatib et al. [EKXY03], Warren/Hutchinson [WH03], and Cárdenas et al. [CS05a] present articles focusing on this interdisciplinary research. However, none of these articles presented results of a threat analysis for this domain. Hence, their contributions will be considered in the following where appropriate to gather recommendations and possible conceptual or technical weaknesses in e-learning systems. For a closer definition of still open research questions, relevant monographs are examined. With respect to available literature, only two monographs can be found matching stated demands, i.e.,

[1]Before 2003 Kluwer Academic Publisher was in this position. In 2003 Kluwer merged with Springer to Springer Business+Science.

the dissertation thesis of Graf [Gra03] and the book of Weippl [Wei05]. These documents will be discussed in the following.

In his thesis, Graf [Gra03] aims at specific security mechanisms for learning environments. For this, he started his work with a short consideration of learning related aspects like the learning process or the general construction of learning material. The "learning process" according to his thesis includes the access to learning resources, the actual learning activity, assessments, and personal guidance for advising next steps [Gra03, p. 6]. In addition to basic concepts, he stated problems related to web-based client-server architectures for deploying an e-learning system, e.g., the lack of persistent data storage due to missing state information or identification mechanisms in HTTP. Note that Graf explicitly prefers the use of client-server architectures over stand-alone clients, since by this distribution sensitive processes can be put on server-side where protective measures can be applied and controlled more reliably [Gra03, p. 16]. An analysis of the threat situation for e-learning is conducted in his Chapter 6 (10 pages). This chapter includes the identification of assets, the identification of threats, and the rating of found threats. Obviously, with respect to this small number of pages for the analysis it can be considered to be kept very superficial. Subsequent chapters present and discuss possible solutions to mentioned threats. In addition, for the entire discussion within his thesis, Graf explicitly excludes communication and cooperation [Gra03, p. 2 and p. 8]. Learning in groups, as well as the use of computers as communication medium, are mostly neglected, as are the management of personal data.

Graf contributes to this research by presenting learning specific security aspects [Gra03, pp. 95-97], i.e., confidentiality of learning material and assessment results, authenticity of user activities, integrity of learning material and assessment data, accountability, and copyright protection. Solutions as discussed by Graf include encryption enforcement of learning material [Gra03, pp. 123-126], i.e., all learning contents are stored in an encrypted way on the server and client, such that no direct access to unencrypted material is possible for (authenticated) persons or persons gaining access to these data by other means, e.g., stealing storage media. Encrypted data can only be decrypted again by an auxiliary program connecting to the server and requesting appropriate keys. This even provides improved ways for learner surveillance due to possibilities for recording every access request to learning contents. Note that such an encryption enforcement is usually only advantageous for content providers and may be very distracting and limiting for (honest) learners. For the aspect of copyright protection, Graf proposes the integration of digital watermarks to identify the last authorised requester [Gra03, pp. 126-131]. Although this still does not allow the prevention of unauthorised distribution of material, it enables the extraction of the last legal user who most probably started the unauthorised distribution process. A major topic of Graf's thesis is given by web-based assessment systems [Gra03, pp. 141-166]. Graf developed a framework for enabling assessments in controlled environments. Although a monitor is still necessary during assessments, the framework basing on Java applets on client-side and Java applications on server-side takes care of fair and equal time constraints, such that no learner has more or less time than another learner for doing an exam. In addition, interim results are always transmitted to the server for providing fault tolerance in case of client errors that may make it necessary to change the client machine.

Consequently, with respect to the short analysis phase that, in fact, raises questions concerning the source of identified threats, Graf clearly puts the focus on a set of solution proposals. This contributes to the research field of security in e-learning, but the base problem of a clear analysis of possible threats in e-learning is not solved satisfactorily.

Weippl [Wei05], on the other hand, aims at an overview of research topic and divides his book in three parts separating a superficial quick start guide, an in-depth consideration, and additional resources. In the first part, Weippl considers role specific expectations and related security demands. As roles for this consideration he proposes authors, teachers, managers, and students. Authors, in his opinion, are mainly focusing on the confidentiality and integrity of learning contents, including basic assets like images and texts. Teachers would mainly demand security of course contents or online exams. Course contents in this context refer to communication data that are possibly stored for several years using long-term data retention, e.g., in the form of backup tapes, such that privacy aspects are affected. If these tapes are not protected sufficiently or if data are even stored without proper protection within the system itself, then messages could get public and harm confidentiality demands. Managers in [Wei05] represent executives of an institution whose demands are concerned with legal regimentation, business continuity, and prevention of negative publicity. Finally, for students' concerns, he mentions possibly unauthorised read access to their personal notes, as well as exhaustive logging mechanisms recording every single activity of students. For none of the mentioned roles he does provide a satisfactory threat analysis.

In the in-depth part Weippl discusses technical security aspects including privilege management, digital watermarking for protecting images, and cryptographic means. For this, he significantly moves the focus from learning related aspects to pure technical issues. Although Weippl integrates a security risk analysis chapter [Wei05, pp. 73-96], he does not actually conduct an analysis, but only presents different approaches for doing this. Even subsequent chapters do not pick up this missing point. Instead, general aspects like access control and organisational strategies, e.g., for backups or deletion of files, are considered. Additional resources, as presented in the third part, also do not supply a collection of e-learning specific risks, but only present encryption mechanisms basing on PGP (Pretty Good Privacy) and exemplary plagiarism detection services. Thus, Weippl also does not provide a systematic threat analysis for the domain of e-learning. He only presents a set of general and less e-learning specific security issues and counter-measures. Due to this general description, a transfer to specific e-learning scenarios is not possible.

Hence, both monographs as considered in this section did not describe a systematic threat analysis. Also, none of these works introduced a well-founded collection of specific threats. This still is open for further research. In fact, this thesis is aiming towards such an analysis to provide an appropriate base for a subsequent investigation concerning the creation of counter-measures. The remainder of this chapter, therefore, will be used to investigate related literature and extract respective expectations, requirements, and base conditions for actually conducting a satisfactory and systematic threat analysis to fill the discovered research gap.

2.4 Terminology and Structure for Extracting Activities

The remainder of this chapter aims at extracting relevant activities and requirements from all affected disciplines. Obviously, this includes the disciplines of information security as part of informatics, and educational science for learning specific aspects. Since e-learning systems furthermore include software components, software engineering also is reasonable to explicitly be considered for an appropriate software design and related implications on e-learning systems. This aims at the integration of well-known technical tasks for web-based systems and is meant to bring these aspects into relation with particular demands for e-learning. To structure the collection of relevant tasks and activities, a denomination for the three investigated research areas (with focus on e-learning) will be used as follows: ($\mathcal{A}$) educational science, ($\mathcal{B}$) software engineering, and ($\mathcal{C}$) information security. All of these research areas will be considered as possible sources of related activities in e-learning. This also includes possibly contradictory activities, as well as mutually implied ones. Thus, e-learning systems will be analysed from three different starting points. Results will be brought together and harmonised in subsequent chapters. Note that due to the importance of educational science for an appropriate consideration of learning related requirements, class $\mathcal{A}$ will be further divided into five subclasses $\mathcal{A}_1$–$\mathcal{A}_5$, each representing a design criteria to be described in Section 2.5. As a second level denomination, $\mathcal{A}$–$\mathcal{C}$ will be further divided in order to separate organisational human-centred approaches from more technical ones.

This separation is meant to simplify the transfer of identified activities to manageable aspects due to a pre-clustering in abstract human-related or already concrete technical activities.

Combining these two levels and including three reasonable subcategories for each main class, this results in a structure as follows:

$\mathcal{A}_x$: educational science ($x \in \{1, \ldots, 5\}$, see above)
- $\mathcal{A}_{x,1}$ learning process
- $\mathcal{A}_{x,2}$ learning support
- $\mathcal{A}_{x,3}$ learning management

$\mathcal{B}$: software engineering
- $\mathcal{B}_1$ usability
- $\mathcal{B}_2$ functionality
- $\mathcal{B}_3$ architecture

$\mathcal{C}$: information security
- $\mathcal{C}_1$ human factor
- $\mathcal{C}_2$ program logic
- $\mathcal{C}_3$ infrastructure

At the end of this chapter, a collection of tasks and activities in e-learning systems will be presented. To explicitly mention the addition of a specific activity to some collection, activities will be highlighted using the following terminology: ($\rightarrow \mathcal{X}_{[a],n,m}$: $\langle name \rangle$). In this example, task "$\langle$name$\rangle$" will be added to the n^{th} subclass of class $\mathcal{X}$, with $\mathcal{X} \in \{\mathcal{A},\mathcal{B},\mathcal{C}\}$ and $n \in \{1, 2, 3\}$ according to the given itemisation. Index m is used as counter to uniquely distinguish the tasks, and therefore, to allow clear referencing for subsequent discussions. The special case of using an additional index a only applies for $\mathcal{A}_1$–$\mathcal{A}_5$ with $a \in \{1, \ldots, 5\}$.

2.5 Implications from Educational Science for E-Learning Systems

As indicated in Section 2.2, the focus for e-learning systems "in the past has been on the 'e' [...]" [Ham02]. But this, obviously, is suboptimal with respect to an adequate conceptualisation and implementation of e-learning as also supported by the OECD (Organisation for Economic Co-operation and Development):

> "Technology alone does not deliver educational success. It only becomes valuable in education if learners and teachers can do something useful with it." [OEC01, pp. 24-25]

What "useful" can mean in the context of e-learning will be closer investigated in this section considering literature primarily focusing on the learning process and related requirements. The educational scientist Lipponen explicitly emphasises learners as a special target group for software engineers, such that very specific activities have to be implemented and usability should be adequate to support learning:

> "Learners are not the same as the everyday people or experts, but need software designed especially for the learners." [Lip02, p. 78]

Schulmeister criticises in [Sch03b] that most evaluation catalogues to rate and test e-learning systems are technology-driven. He claims this specific development focus without sufficiently taking care of didactic aspects to be a "historical step backwards" [Sch03b, p. 151]. Consequently, he took part in a project called EVA:LERN with the aim of creating an evaluation catalogue that focuses on didactic aspects. For this, 23 international studies were merged, containing together

876 criteria (including overlaps). This enormous number was reduced to 181 criteria best fitting to their notion of a didactically driven evaluation (cf. [Sch03b]). An excerpt of these criteria will be used in the following.

Eibl and Schubert introduced six design criteria for classifying educational science implications for e-learning [ES08a]. In the following, five of these criteria will be applied as structure for the investigation of relevant activities. The sixth criterion "equal opportunities" will not be considered in this place due to the lack of directly connected activities – despite its pervasive presence and relevance in e-learning. However, it will be included again in Chapter 3 for discussing assets basing on results of this chapter. Note that this classification with design criteria is independent of the type of learning, i.e., self-study versus guided teaching. Thus, it can be applied easily to respective e-learning scenarios:

$\mathcal{A}_1$) social support by cooperation and communication,

$\mathcal{A}_2$) activities of students as important step of the learning process,

$\mathcal{A}_3$) priority to meet learning objectives,

$\mathcal{A}_4$) flexible learning,

$\mathcal{A}_5$) integration into learning environment.

These criteria $\mathcal{A}_1$ to $\mathcal{A}_5$ will be used to structure the remainder of this section.

2.5.1 Social Support by Cooperation and Communication

Communication is an essential part of e-learning, since it enables discussions and agreements as well as negotiation of common understanding [Sch03b]. Furthermore, it can foster soft skills like group organisation or team work [Ker01, p. 281], and can have positive influence on learner's activity due to accountability to the group [Sta01, p. 46].

Kerres distinguishes three kinds of communication and cooperation [Ker01, p. 266]:

- informal exchange of information, i.e., persons communicate for the sake of social contacts and not necessarily related to learning topics,
- project related collaboration, i.e., learners work together on the same project and try to solve problems as community, and
- helpful cooperation, i.e., persons work on different projects but support each other with their individual knowledge.

This emphasises differences in e-learning systems between simple communication facilities to allow social contacts and cooperative systems to explicitly work on assigned projects. However, note that from a technical point of view the exchange of information for an informal communication is not different to communication helping other learners with their specific problems. Wessner presented a division that actually provides three, even technically different forms of cooperation [Wes05, pp. 101-102]:

- generic cooperation, i.e., cooperation and every form of communication that is not bound to a course and that is not necessarily related to learning contents ($\rightarrow \mathcal{A}_{1,1,1}$: *generically communicate*),
- spontaneous cooperation, i.e., getting in contact with other learners or teachers with the aim of receiving support within a course for a specific learning topic (cf. Sec. 2.5.4),
- intended cooperation, i.e., planned events during a course for letting learners work cooperatively on a given problem ($\rightarrow \mathcal{A}_{1,1,2}$: *participate in intended cooperation*).

In general, for communication and cooperation, in the evaluation catalogue presented by Schulmeister [Sch03b] whiteboard support, forums, and chat were mentioned as must-have criteria. Other systems like e-mail or messenger systems are only marked desirable. Application sharing is marked optional.

Communication in principle has two aims: First, it is a psychological need to get and stay in contact with others [Lin77], and second, mutual assistance [BBR62, WFS02] by expressing one's own understanding and by discussing it with other peers:

> "Many educators and learning researchers have found promise in Vygotsky's (1978) social constructivist ideas about learning in a social setting. His notion of the zone of proximal development (ZPD) posits that children's mental development can be positively influenced by the assistance of an adult or more capable peer." [WFS02, p. 245] (referencing [Vyg78])

Although mutual assistance is possible by already mentioned communication facilities, it provides an additional task not to be confused with informal generic communication ($\rightarrow \mathcal{A}_{1,1,3}$: *mutually assist each other*).

With respect to intended cooperation, the management of groups is required. This includes properties of groups for adequately being able to work within the system and properties of learners for creating groups of compatible members. According to Wessner groups can be composed by learners, teachers, random, or automatically following certain patterns [Wes05, p. 29]. Haake et al. also identified "four typical forms of group composition" that should be supported by cooperative learning platforms:

> "(1) explicit assignment of members to a group, (2) invitation to a group, (3) free access to a group and (4) confirmed access to a group after a confirmation of the group itself" [HBH$^+$04, pp. 235-236] (translated from German)

They claim that all these modes should be available to the end user, since only the users themselves would be able to define their group desires and to choose the best way of creating the group. With respect to tasks of users in e-learning, only the former two methods as given by Wessner will be regarded in the following ($\rightarrow \mathcal{A}_{1,1,4}$: *manage groups (as teacher)*)($\rightarrow \mathcal{A}_{1,3,1}$: *join group (as learner)*). However, the confirmation aspect will be kept in mind for the recommendation part.

A major task for teaching staff, including mentors and tutors, is the support of learners. This is especially of relevance in distant courses, since no contact in person is given, and as such, disappointments and cognitive problems cannot be recognised in the same manner as in presence teaching. Communication styles and usage of computers as medium for communication are also affected:

> "Left to themselves learners might be reluctant to disagree or challenge [...] Tutors need to encourage divergence within the group, suggest roles, introduce 'starter' and 'wrapper' activities and develop strategies to compensate for lack of non verbal and paralinguistic clues." [HW05, p. 3]

Schulmeister even mentions perpetual support in learning as a possible solution to mitigate cheating, since a close contact between learners and teaching staff will reduce the probability of not knowing the learners' capabilities and interim solutions ($\rightarrow \mathcal{A}_{1,2,1}$: *mentor learners in case of problems*). Hence, plagiarism and cheating attempts can be detected more easily and will therefore be less likely to occur (cf. [Sch03b, p. 166]). In addition to supporting learners, Wessner emphasises the role of a moderator for providing increased efficiency in cooperative processes and for ensuring fairness during group works. Main tasks of moderators are management of groups, providing auxiliary material, explaining assignments, observing interactions, intervening for supporting the application of cooperative methods and social competences, and finally the evaluation and reflection of the learning process within the group [Wes05, pp. 106-108]. These subtasks will be subsumed to the main task of moderating learning group cooperation ($\rightarrow \mathcal{A}_{1,2,2}$: *moderate cooperation*).

A psychological study concerning the effects of awareness aspects and member portraits in cooperative systems is given by Cress [Cre05]. Awareness ("Who is there?") often is considered to be necessary for appropriate cooperation (cf. [HB03]). Furthermore, member portraits ("What person is behind this account?") are considered to be helpful for creating a certain level of accountability to the group (cf. [Sta01]). However, Cress stated this bipartite hypothesis to reflect these common, general assumptions:

> "Based on the [Social Identity Model of Deindividuation Effect], we expect that providing member portraits will lead to a higher number of contributions for people with individualistic orientation individuals, but to a lower number of contributions for people with prosocial orientation, revealing an ambivalent effect of member portraits as a group-awareness." [Cre05, p. 88]

Using a psychological placement test, the majority of the considered test persons was recognised to be prosocial. In her study she could verify this hypothesis, such that in most cases, negative effects could be more likely than positive effects. Member portraits as found in most practically used e-learning systems, therefore, will not be considered to be mandatory in this thesis. Nevertheless, their management will be considered for the sake of completeness in Section 2.5.5.

2.5.2 Activities of Students as Important Step of the Learning Process

Due to the change in learning theories, the importance of interactivity for e-learning systems increased significantly. To enable a high degree of activity of learners with the system, interaction possibilities must be available to a large extent. Interactivity is necessary especially for reasons of efficiency of learning, i.e., being able to actively deal with learning material instead of only passively listening [FU04]. But note that interactivity can be seen from different perspectives. While Geer states that

> "Interactivity can be defined both in terms of social and content interaction." [GB01, p. 250]

she mostly considers communicative interaction forms like different kinds of discussions and collaboration as already discussed in the former subsection. Schulmeister, on the other hand, explicitly states that interactivity in e-learning is not aiming to social interaction but to the manipulation and usage of learning objects for learning purposes [Sch03b, p. 156]. In the following, we will use the notion as given by Schulmeister.

Concerning the actually relevant level of interactivity in an e-learning system, no common sense is available, i.e., interactivity can be regarded in different levels, all to be integrated in certain degrees. Schulmeister [Sch02] introduced six interactivity levels that increasingly provide more possibilities of interacting with learning material and the e-learning system (cf. Tab. 2.1). Especially noteworthy are the transitions from level III to IV as well as from V to VI. Levels I-III do not allow modification of contents, but at most enable the modification of the content representation (level III). In level IV, interaction forms are considered that allow adapting and influencing learning content. The latter mentioned transition adds the aspect of automated reaction to learner's input, such that the system can give feedback and acts as "partner" in the learning process. The subdivision of Schulmeister was reduced by Schwidrowski et al. to four practically better manageable and robustly applicable levels of interactivity [SES08, pp. 147-148]:

1. communication among humans using informatics systems,
2. interaction with informatics systems without changing learning objects,
3. interaction with informatics systems with influencing learning objects,
4. interaction with informatics systems including learner related feedback.

While the first two levels are considered for other design criteria (cf. Sec. 2.5.1, 2.5.5), the criterion of activity of learners is mostly related to the latter mentioned levels. This means, learners must

Table 2.1: Taxonomy of multimedia components [Sch02] (levels translated from German)

	level	description	examples
I.	regard and receive objects	no influence on (re-)presentation of components	view images or play back audio/video files
II.	regard and receive multiple presentations	different static view on same topic; no change of data	choose object from set of predefined ones, pass through series of data
III.	vary representation form	influence representation of dynamic data; no change of data	turn and zoom of 3D objects
IV.	modify content of component	creation and modification of content in certain parameters	simulations with changeable parameters
V.	construct object or content of representation	apply tools for object or model construction	construct geometrical figure, create mindmap
VI.	construct object or content of representation and receive intelligent feedback from system by manipulating actions	user input gets evaluated and influences learning process	mathematical editors

get the possibilities to work with learning objects and influence their behaviour, e.g., to process simulations with own entered values or to interactively influence further steps of an animation ($\rightarrow \mathcal{A}_{2,1,1}$: *interact with influencing learning contents*). In addition, Kerres and Schulmeister put emphasis on feedback mechanisms to provide learners with information about their learning progress and possible problems. Note that this is not meant in the sense of so-called intelligent tutoring systems, but only in the sense of intended switching points using self-assessment for checking the learning progress. Schulmeister [Sch03b] rates interactive tests and exercises as must-have criteria for e-learning, while (formal) self-assessments and possibilities to check one's own learning outcomes are only mentioned to be optional. Kerres states that assessments were most relevant with the notion of behaviouristic learning theories, but lose their relevance with the consideration of more actively characterised learning theories like constructivism [Ker01, p. 200]. Nevertheless, although the creation of (good) tests is time-consuming and expensive, this can provide progress information for learners and can be used to determine or advise further steps in the learning environment [Ker01, p. 201] ($\rightarrow \mathcal{A}_{2,1,2}$: *interact with systems providing feedback*).

Finally, for successfully and adequately passing through a course, learning paths should be defined and evaluated. Schulmeister mentions these steps as (highly valued) desirable criteria [Sch03b] ($\rightarrow \mathcal{A}_{2,3,1}$: *create learning paths*)($\rightarrow \mathcal{A}_{2,2,1}$: *evaluate chosen learning paths*). Related to this, the content aggregation model (CAM) documentation of the SCORM (Sharable Content Object Reference Model) standard describes different kinds of structuring learning content by different views of developers that can be directly transferred to different learning types:

> "A content organisation can be seen as a structured map of learning resources, or a structured activity map to guide the learner through a hierarchy of learning activities that use the learning resources. One content developer may choose to structure the content organisation as a table of contents for the learning resources, while another content developer may choose to structure the content organisation as an adaptive guided path through a learning experience, invoking learning resources only if and when they are needed. A third content developer may create a content organisation where some discovery activities include a free form use of some of the learning resources, while other activities are more formally managed." [Adv09, p. CAM3-8]

The option for learners to choose between sequentially structured courses or to freely discover contents allows better adaptation to the kind of motivation, i.e., intrinsic of extrinsic motivation, and learners' previous knowledge which is related to the probability of being overstrained by too much freedom [Ker01, p. 139].

2.5.3 Priority to Meet Learning Objectives

Learning with books is related to well-known and physically tangible interfaces that can be easily handled by learners, e.g., sequential structure of information, table of contents, or even index pages for finding important aspects. In opposition to this, web-based systems could be perceived as chaotic hypertext structures without revealing the actual amount of data but only showing single pages with hiding the remaining ones. Thus, in this design criteria aspects are in focus that support the orientation within learning contents and available learning paths, and in addition, that provide (technical) support for learners to enable them to get back to the primary task of learning with the system. Thus, technical problems and distracting functionality should vanish and stay hidden in favour of effectively learning with the system. Note that the criterion of keeping the priority on learning objectives in most parts is based on sophisticated and appropriate usability concepts, i.e., only if the system is usable in an adequate manner, learning can take place without much distraction. Nevertheless, not only generic software engineering aspects, but also supportive elements from a less technical perspective are applicable.

Concerning supportive systems directly implemented into the e-learning system, Schulmeister described (generic) online help systems as must-have criterion [Sch03b]. Schulmeister does not demand help systems to be context-sensitive, i.e., it is not required to directly show relevant help but only to offer a system where learners can search for appropriate information. In the EVA:LERN catalogue, context-sensitive online help systems are mentioned as optional with low value of importance. Thus, either one of these variants should be available ($\rightarrow \mathcal{A}_{3,2,1}$: *maintain (context-sensitive) online help*). In addition to help information, Schulmeister considers a collection of common problems and emerging questions to be sensible ($\rightarrow \mathcal{A}_{3,3,1}$: *maintain FAQ*) as well as sitemaps for better orientation within learning paths ($\rightarrow \mathcal{A}_{3,3,2}$: *maintain sitemap*).

Besides technical support systems and prepared help systems, human support can be very valuable in more complex problem situations. Technical systems that cannot be simplified beyond a certain point to enable all possible learners to work with it efficiently introduce requirements for proper support. For such a situation, Schulmeister mentions preparatory education for system related aspects ($\rightarrow \mathcal{A}_{3,2,2}$: *educate system related aspects*). Furthermore, hotline support can provide helpful information in case of urgent situations or problems not solvable by learners on their own ($\rightarrow \mathcal{A}_{3,2,3}$: *provide hotline support*).

Concerning long-term improvement of courses, statistics, log files, and (generic) user tracking are mentioned by Schulmeister [Sch03b]. Kerres seconds this approach of evaluating learning behaviour like duration of learning, number of interrupts, way of using the system, and progress within a course [Ker01, p. 112]. Taking records of user behaviour and possibly adapting system's reaction to previous behaviour is also indicated in the Learning Technology Systems Architecture (LTSA) created by the Institute for Electrical and Electronics Engineers (IEEE) Learning Technology Standards Committee (LTSC) [IEE01]. Note that we have to distinguish (automated) technical tracking ($\rightarrow \mathcal{A}_{3,2,4}$: *track and evaluate user activities*) and explicitly conducted surveys using questionnaires and involving learners who take part in this survey voluntarily ($\rightarrow \mathcal{A}_{3,2,5}$: *conduct evaluation survey*).

2.5.4 Flexible Learning

E-learning systems can be used with different objectives. While most platforms are only "abused" as simple presentation systems managing documents or provide chaotic communication channels (cf. [Sch03b, p. 152]), e-learning systems also can aim at supporting the learning process with as much flexibility as known for presence teaching scenarios. In presence teaching, the process of teaching, besides proper preparation and technical competences of teaching staff, requires capabilities for a flexible adaptation to emerging problems and changing situations [Ker01]. From a technical point of view, Wang mentions "essential features" as follows:

> “Essential features addressing multiplicity and flexibility will be discussed including: multiple course representations, tools, communication types and channels, support, interactions, flexible course structure, personalised communication and feedback system, and cultural-sensitivity to better meet the needs of learners from diverse backgrounds.” [WR02, p. 575]

Thus, flexibility in e-learning systems can be considered in two perspectives. First, flexibility in communication and level of user interactivity, and second, flexibility with respect to course creation as well as integration of intended activities and cooperative elements. Concerning the first perspective, spontaneous cooperation as mentioned above (with reference to Wessner [Wes05, pp. 101-102]) provides flexible access to other (more experienced) peers and tutors for receiving support in currently emerging problems ($\rightarrow \mathcal{A}_{4,2,1}$: *participate in spontaneous cooperation*). With respect to the second perspective on flexibility, Gee et al. stated that it should be an aim to design tools in a way

> “to be flexible so that they can be used at a number of levels, from major pedagogical re-engineering of courses through to enrichment of aspects of the learning process with engaging and illustrative resources.” [GCC$^+$05, p. 2]

This includes generic (re-)engineering of courses to adapt the structure and content according to the current situation including specific target groups ($\rightarrow \mathcal{A}_{4,3,1}$: *(re-)engineer courses*). Furthermore, Gee et al. propagate engaging and illustrative resources. In web-based e-learning systems, this most probably requires the integration of plug-ins to interpret specific file formats for animations with sophisticated program logic and appealing contents. In most cases, these are superior to rather static web pages ($\rightarrow \mathcal{A}_{4,2,2}$: *include illustrative dynamic resources*). Note that with respect to web accessibility (cf. [Bud04]) the use of plug-ins should be avoided due to increased dependencies on client-side and reduced opportunities for persons with inabilities to access information encoded in corresponding (mostly proprietary) file formats. Nevertheless, HTML (hypertext mark-up language) and CSS (cascading style sheets) are not sufficient for providing flexibility concerning a high degree of user interaction, and as such, their exclusive usage cannot be guaranteed for e-learning.

Wessner explicitly states requirements for good collaborative systems [Wes05, pp. 48-50]. In particular, he claims the need for flexible transitions between different learning modes, e.g., individual learning versus synchronous/asynchronous cooperative learning. On an earlier page in his dissertation Wessner explicitly distinguishes four parts of such modes and methodologies [Wes05, p. 15], i.e., articulation of lessons like structuring in phases, kind of process like analytical, synthetic, or project-based process, social forms for dividing learners into small groups or teach them all together, and finally action forms like discussions, demonstrations, silent work, or generic presentations. Ferstl and Ulrich [FU04] second this subdivision by presenting comparable forms. For flexibility in e-learning, mostly transitions between action and social forms are of interest. Kerres additionally divides action forms into presentational, activating, and structuring forms [Ker01, p. 179]. While presentational action forms include multimedia presentations, audio or symbolic information, and visualisation elements for procedures and processes, activating forms include animations with interactions and communicative elements that demand learners to participate in discussions and interact with presented material. Structuring action forms are mostly reduced to meta-information, which allows better navigation and switching of current learning topics ($\rightarrow \mathcal{A}_{4,3,2}$: *use different action forms*). Transitions of social forms are mostly related to capabilities of simply creating or breaking up groups as intended for the course progress with individual and cooperative learning phases. While different ways of composing groups were mentioned in Section 2.5.1, for flexibility of e-learning systems, the integration of and switching between social forms are considered. For this, of course, a specific set of possibilities for creating groups should be available with best flexibility to be integrated into the course and to be best manageable at least by teachers for intended cooperative phases ($\rightarrow \mathcal{A}_{4,3,3}$: *apply different social forms*).

2.5.5 Integration into Learning Environment

Obviously, user interfaces and possibilities for users differ in working with books or paper notes, versus e-learning systems with an informatics system as single interface for managing the presentation of learning material and storage of notes. In this context, one major aspect for the integration of e-learning into a learning environment refers to all tasks and activities to be integrated in e-learning systems in order to bridge the gap between presence teaching and e-learning. This means, presence teaching with decentralised note taking and learning with paper-based material, such that annotations can easily be applied, needs to be transferred to e-learning appropriately. With respect to space and time independence for learners, e-learning systems have to centrally store all relevant data for providing increased flexibility. Thus, every learner spontaneously can connect over a nearby computer and can access all of his notes and materials. A second aspect touching this integration is given by connections to foreign systems, such that a single portal can be provided for the learner while all other system components are encapsulated and hidden behind a single user interface. This aspect is rather related to an architectural view with respect to software engineering, and therefore, will be further discussed in Section 2.6.1.

Starting from the very beginning, learners browse offered contents and navigate through the material. This was already introduced as interaction of very low level, i.e., interaction without changing learning contents (cf. [SES08]) ($\rightarrow \mathcal{A}_{5,1,1}$: *interact without changing learning contents*). Activities that are well-known for traditional learning and hence should be transferred to e-learning are the management of bookmarks ($\rightarrow \mathcal{A}_{5,1,2}$: *manage bookmarks*), the management of personal notes including sharing of notes with other learners ($\rightarrow \mathcal{A}_{5,1,3}$: *manage personal notes*), and the management of annotations ($\rightarrow \mathcal{A}_{5,1,4}$: *manage annotations*). All these items were explicitly listed in the catalogue presented by Schulmeister [Sch03b]. Kerres particularly encourages the use of annotations, since writing annotations to learning material supports the creation of cognitive structures which is not given in the same manner with passively perceiving other media like films [Ker01, p. 248]. With respect to cooperation as discussed in Section 2.5.1, user profiles and portraits can be advantageous for individual learners to feel more comfortable by knowing at least some more details about possible communication partners ($\rightarrow \mathcal{A}_{5,2,1}$: *maintain user profile and portrait*). This in particular may be desirable for some learners to voluntarily disclose cognitive problems to teachers who are not known in person. Sharing intimate and possibly embarrassing facts most probably will be easier for learners if they feel comfortable and are familiar with a communication partner. As a final task, we consider the management of course participation ($\rightarrow \mathcal{A}_{5,3,1}$: *manage course participation*). While this is rather obvious in presence teaching, it explicitly has to be integrated in e-learning systems. This means, teachers should be able to manage a list of participants fulfilling preliminary course requirements and learners, on the other hand, should be able to enrol in courses they are interested in. The latter case especially is of relevance with respect to commercial courses, such that participation also affects billing mechanisms. From the point of view of orientation and status accessibility, a list of courses where a learner is enrolled should easily be available, e.g., in form of a starting screen ("portal", cf. [Sch03b]) that even could be used to switch courses.

2.6 Security Engineering for E-Learning

Having considered tasks and activities from an educational science perspective, this section will consider e-learning systems from the perspective of informatics, in particular from the viewpoints of software engineering and information security. This includes relevant criteria for software quality and modularisation concepts, e.g., a separation of modules using implemented interfaces or service-oriented architectures. Also, features in e-learning systems for supporting learners will be regarded, besides directly learning related tasks. In addition, due to the increased complexity with respect to constructivist learning theories, it is crucial to construct an e-learning system as simple and modular as possible, such that deficiencies can be easily recognised and possible security incidents be prevented:

> "There are two ways of constructing a software design. One way is to make it so simple that there are obviously no deficiencies and the other is to make it so complicated that there are no obvious deficiencies." (Charles Antony Richard Hoare, 1980 at ACM Turing Award Lecture)

Including all educational science requirements into the design, a very simple architecture is not sufficient for supporting the learning process as a whole. Hence, an appropriate e-learning system must be designed in a way that security is an inherent property, such that the non-obvious deficiencies as mentioned by Hoare can be mostly eliminated instead of simply hiding them from intruders.

In the first part of this section software architectural aspects will be considered, and in the second part information security tasks will be added accordingly.

2.6.1 Software Architecture

The primary expectation of users, although mostly implicit, is the reliability of used software and the ease of use. This in particular is necessary in e-learning with respect to avoidance of distraction by malfunction or unreachable services:

> "Learners need reliable access and user friendly tools." [HW05, p. 3]

Examples for reliability in error-prone procedures are given by Caeriro-Rodríguez et al. [CAL05] who particularly mention concurrency and session controls. Thus, collaboration with using the same resources must be prepared for ensuring data consistency despite concurrent access.

A possible means for providing reliability is given by modularisation and the reuse of practically approved modules. Increased reusability of components is advantageous due to a simplified exchange of single modules without directly influencing other modules as far as interfaces were respected accordingly. Hence, instead of rewriting modules or course data anew every time they are needed, which is an error-prone process, already existing and reliable modules can be used. Concerning activities in e-learning systems this is mostly related to the management of metadata for assets and learning objects ($\rightarrow \mathcal{B}_{3,1}$: *manage metadata*). In e-learning, reusability can be regarded for different components. On the one hand, the reuse of learning contents and didactic concepts, e.g., course structures and course concepts, should be possible. Note that the transfer of didactic deliberations and concepts is much more difficult than a simple reuse of multimedia assets like single videos or graphic files. On the other hand, activity and information modules of the e-learning system are subject to reusability. This includes personalisation, i.e., the use of an appropriate and minimal set of modules a user wants to access [SSK03, p. 225].

Concerning reusability of software components and modules, the Open Knowledge Initiative (OKI) has to be mentioned. OKI aims at creating language independent interface specifications to ensure appropriate modularisation and a simplified exchange of modules within an e-learning system. In its basic aims OKI is very promising and versatile:

> "The Open Knowledge Initiative (O.K.I.) is defining an architecture that precisely specifies how the components of a learning technology environment communicate with each other and with other campus systems. By clearly defining points of interoperability, the architecture allows the components of a complex learning environment to be developed and updated independently of each other." [Edu02, p. 1]

But in fact, with respect to published specifications, OKI is strongly focusing on technical implementations, i.e., interfaces of single modules are defined for classes in an object-oriented architecture. Major criticisms on OKI specifications are, on the one hand, the lack of functionality, i.e., only very basic functionality for e-learning situations is considered, and, on the other hand, additional wrapper functions are required for certain tasks in order to encapsulate basic system calls. Especially the latter aspect, although desirable from a software engineering perspective, breaks the conformity to OKI [Zeu08]. Furthermore, an OKI-based system after deployment appears as monolithic application (cf. [SSK03, Zeu08]), such that modularity is no longer given at run-time.

Note that more recent documents published by core developers of the OKI specifications, e.g., Coppeto [CS05b], propagate OKI to be seen as a service-oriented architecture (SOA), which represents the current state of the art in architectural design. Service-oriented architectures describe informatics systems in a more abstract way by only specifying services to be available using defined interfaces – even if modules are distributed over the network. With SOA, a more abstract view on relevant services is followed, and the focus is on flexibility due to missing dependencies among these services. Note that the technical architecture and even the kind of technologies used for realising such services are explicitly neglected for the SOA concept. Technological details are only relevant for members of the technical staff who actually implement a service. Krafzig et al. mention this as "decoupling of functionality and technology" [KBS05, p. 7] which, in fact, simplifies porting of single services to other platforms and provide exchangeability of services as far as interfaces are kept consistent.

Lehsten et al. [LTZ+08] outlined basic architectures, e.g., client-server-architecture, and claimed that broker architectures can best solve emerging problems concerning appropriate flexibility for the use of modules. Several services could be distributed over the network and be made available using corresponding interfaces ($\rightarrow \mathcal{B}_{3,2}$: *distribute and connect to services*) to compose a proper architecture including all relevant functional services, e.g., management capabilities, communication and cooperation facilities, and also mediating components for working with standards ($\rightarrow \mathcal{B}_{2,1}$: *provide and enable proper functionality*). Connected services do not need to be visible in the front-end application, which by Schulmeister is called "portal" [Sch03b, p. 143]. But this portal can encapsulate and manage connections to external services ($\rightarrow \mathcal{B}_{2,2}$: *manage connections to external services*). The broker service as mediator can be used for searching for appropriate services, while the services themselves register with that broker in advance. For example, a library search system (cf. [Sch03b]) providing an overview of all topic related literature could be integrated into the e-learning environment. In this thesis, access to external services will be considered in a generic manner without any limitation to a specific set of services, such that the e-learning system can act as global system spanning several components, and therefore, leads to an integrated learning environment [FU04, p. 105].

Another major aspect for e-learning systems is given by the adaptability to specific situations (cf. [Sch03b]). This includes technical aspects like proper font and character encoding, e.g., Unicode support, to enable learners to spell their names with the correct characters, and also to allow them to communicate in their native language without using replacement strings [Jev06]. Concerning different application fields and areas of study, various functional requirements could be stated, e.g., natural sciences and mathematics usually need formulae for their content presentations, such that input filters might be necessary to support easy formula integration, e.g., in LaTeX syntax. With respect to further specialities of the variety of study courses, e-learning systems, obviously, should be easily extensible ($\rightarrow \mathcal{B}_{3,3}$: *extend system*) and modules should be manageable in a way that they can be activated, deactivated, or even be replaced by other ones just as needed for the actual learning scenario ($\rightarrow \mathcal{B}_{3,4}$: *manage modules*).

In addition to technical adaptations by including required modules, more organisational adaptations are desired. This includes the use of a globally defined appearance like a corporate design with templates clearly identifying the hosting company [Sch03b], but also adaptability of per-course designs with respect to the learning group ($\rightarrow \mathcal{B}_{1,1}$: *manage designs*). Furthermore, adaptability also can include personal adaptations to the appearance of a system ($\rightarrow \mathcal{B}_{1,2}$: *per-*

sonalise appearance). User-centred personalisation can support learners by providing a system that is optimally adjusted to individual preferences [SSK03], e.g., almost empty desktop versus "chaotic" layout of all possibly interesting data. Learners in presence scenarios obviously differ in their preferences, and so an e-learning system also should offer functionality to arrange their data accordingly. Furthermore, appearance in this context includes the management of sessions, i.e., if a learner suspends the system and wants to go on later, then the system has to resume from the position where this learner stopped the last time. Such a mechanism does not only guarantee increased comfort for learners but also supports efficiency in the learning progress.

Functionality related to efficient learning is given by a possibility to register user accounts online without the need of involving administrators to enter all relevant data [Sch03b]. Note that administrative influence could still be desired in certain environments to confirm account creation ($\rightarrow \mathcal{B}_{2,3}$: *register user account online*), e.g., for commercial courses. With respect to communication and cooperation, an archive of messages can be helpful to recall former messages in order to prevent multiple similar discussions ($\rightarrow \mathcal{B}_{1,3}$: *archive text messages*). Stahl explicitly mentions storage and archiving capabilities of computers as an advantage in the context of cooperation:

> "[Computers] can overcome the limitations of human short-term memories and of paper-based aides to generating or sharing drafts of documents." [Sta02, p. 1]

Finally, to manage a possibly huge amount of information and to find required data as quickly as possible, the implementation of a search engine is mentioned by Schulmeister as a desired function with high value of importance [Sch03b]. Since this is only the demand for a feature and not to be considered as regular activity in an e-learning system, the integration of search engine capabilities will not be added to the task collection.

2.6.2 Security Concepts

With respect to space and time independence, as often advertised for e-learning, legitimate learners should be enabled to connect whenever and from wherever they like. Consequently, "nomadic users" as described by Sánchez et al. [SLM06] must not be restrained to a specific set of client machines, for example, by applying IP (internet protocol) address filters only accepting some known patterns ($\rightarrow \mathcal{C}_{3,1}$: *change client computer*). Furthermore, multi-user systems might be involved, such that several users could work from the same client computer. This leads to special treatments with respect to web session management and storage of sensitive data on clients. According to Warren and Hutchinson [WH03] e-learning systems should provide automated logout functionality (with destroying web sessions) to ensure that especially in case of computers that are used by several persons no one can access data of a former user. In addition, they claim that the complexity of such systems can lead to misconfiguration, which implicitly suggests that systems even have to be (re-)configured at run-time when already practically used ($\rightarrow \mathcal{C}_{1,1}$: *configure system*).

A basic concept for security is given by user accounts and fine-grained privileges to be assigned to users. With respect to global system configurations, therefore, administrators must be able to manage user accounts that were, for example, registered by users themselves as mentioned above ($\rightarrow \mathcal{C}_{2,1}$: *manage user accounts*). Basing on these accounts, users have to authenticate before being allowed to work with the system. If a user can successfully be identified, the corresponding set of privileges will be granted ($\rightarrow \mathcal{C}_{2,2}$: *authenticate users*). For practical implementation, Gelbord suggests the authentication process to be realised by the use of a PKI (public key infrastructure) to clearly identify learners with more sophisticated mechanisms than simple password-based authentication. Another advantage of a PKI is the integration of non-repudiation mechanisms. If sensitive processes are signed by the user's private key, then authenticity and integrity can be verified ($\rightarrow \mathcal{C}_{2,3}$: *deploy PKI*).

For the management of user privileges ($\rightarrow \mathcal{C}_{2,4}$: *manage privileges*), in e-learning there appears to be an impressive consensus for using role-based access controls ($\rightarrow \mathcal{C}_{2,5}$: *apply roles-based access control*) (cf. [Sch03b, KR04, Wei05]). But note that affiliated roles possibly follow different points of view, and therefore, conflicts in their expectations and behaviour could lead to security problems that need to be managed accordingly [Wei05]. Kienle and Ritterskamp [KR04] explicitly state that the roles of moderators and participants must be separated to not provoke interest conflicts possibly leading to abuse of privileges ($\rightarrow \mathcal{C}_{2,6}$: *clearly separate duties*). Furthermore, role-based access controls must be applied with possibilities to separate subgroups within a specific role [Sch03b], e.g., for cooperative scenarios different groups of learners (all assigned to the same role "learner") should work with only having the privileges to access their own group's results.

Despite all technical security means, users usually are the weakest part of the security concept. This means, fallibility and obliviousness of users can lead to more problems than misconfiguration of systems. Kajava and colleagues [Kaj03] (including [KV02a, KV02b]) investigated security demands for e-learning mostly focusing on users of the system, i.e., how to teach learners to act according to policies in order to not support security breaches by circumventing security mechanisms. Related to security education, security policies can provide helpful information for users on how to act correctly. Note that we have to distinguish formal and informal security policies. Informal security policies address general rules concerning human behaviour, but cannot be evaluated automatically by the system due to their lack of formalised rules ($\rightarrow \mathcal{C}_{1,2}$: *state (informal) security policies*). Formal policies, on the other hand, contain mostly technically oriented rules that can be formalised, and as such, be automatically evaluated and enforced ($\rightarrow \mathcal{C}_{2,7}$: *state and enforce (formal) security policies*). For example, such formal rules can demand passwords to fulfil certain parameters, or state limits on how many and what sort of connections are allowed for some server (cf. intrusion detection systems, [CS05a]).

A particular aspect mentioned by Graf [Gra03] is the content protection with respect to copyright enforcement. Graf describes mechanisms for copyright protection and tracing approaches using digital watermarking, such that information about the last legitimate request for these data is stored within the multimedia asset, and therefore, illegal copies can be traced back to the last requester who most probably will be the one who distributed the content ($\rightarrow \mathcal{C}_{1,3}$: *apply content protection*).

Furthermore, Hentea et al. [HSP03] claimed assessments to be a main topic for e-learning. Although in this thesis online assessments will not be considered (cf. Sec. 1.4.3), intended cooperation and collaborative tasks among groups of learners also require the consideration of mechanisms for preventing and detecting cheating attempts between different groups ($\rightarrow \mathcal{C}_{1,4}$: *prevent and detect cheating*).

Another special case for requirements and activities is given by increased privacy demands. Borcea et al. [BDF$^+$06] present a system managing different so-called partial identities for a single user in e-learning systems, such that each identity can be used for different tasks in a system or for different courses. This is desirable in their opinion to explicitly decouple the situation of being a learner in some course and of being a moderator in another course. No one should have any disadvantage or be treated negatively, for example, by other learners only because of acting in the role of a tutor or moderator in another course ($\rightarrow \mathcal{C}_{2,8}$: *support privacy concepts*). Note that such a concept demands increased privacy and anonymity deliberations on a lower layer. By the use of anonymous communication channels that are managed by a middleware, even the e-learning system cannot be exploited to trace a learner and to link different partial identities to observe users [BDF$^+$06] ($\rightarrow \mathcal{C}_{3,2}$: *ensure anonymous communication*). El-Khatib et al. [EKXY03] investigate similar aspects for e-learning in the form of traffic flow analysis prevention by Onion Routing [RSG97, GRS99] and Mixes [Cha81], such that learners entirely can stay anonymous to the e-learning system. This might be necessary for education within companies, where possibly data protection legislation is affected, if learners can be observed excessively.

2.7 Resulting Limitations for Practical Systems

Besides the sets of activities connected to e-learning scenarios, discussed requirements and expectations of learners also introduce certain limits for security implementations. These limits must be respected in order to receive acceptance on the learners' side, which is essential for actually improving the security situation in e-learning. Obviously, security mechanisms are only of effect if practically used and not ignored or circumvented by users.

Limits and related consequences will be discussed in more detail in the following subsections. In addition, they will be picked up for the creation of recommendations in Chapter 6 to provide suitability of recommendations for the actual demands and expectations connected to e-learning scenarios.

2.7.1 Complexity and Character of E-Learning Systems

E-learning systems in this thesis are understood as (compounds of) applications supporting the learning process as broadly as possible. In opposition to wider definitions that even include the simplest applications dedicated to learning or training very specific matter, e.g., vocabulary trainers, this results in significantly more complex systems. With respect to discussed educational aspects and implications, as given for providing support for the whole learning process, this results in functionality to be included for implementing almost all of the mentioned aspects. Constructivist learning theories result in demands for a high degree of freedom including sophisticated evaluation mechanisms and implementations that cannot be solved by static web-based systems. Furthermore, increasing the interactivity level raises the demand for integrating dynamic elements like using the AJAX (asynchronous Javascript and XML (extensible mark-up language)) approach, stand-alone programs, applets to be loaded within a web page, or animations integrated over plug-in extensions. In addition, flexibility on teacher's choice is desired to enable an appropriate organisation of the group of learners and to manage them in different subgroups with specific tasks. It is sensible to integrate switching points where learners can choose between different presentation forms, exemplary case studies, or intensity levels, e.g., beginner vs. expert level. This demands a sophisticated management of resources to enable contents of different learning paths concerning the same topic to be related internally, such that at the next switching point learners of all learning paths again are brought together to the same thread.

Modularity, portability, and platform independence are key concepts for successful reusability of software components [Bal00]. Especially modularity is considered valuable to reduce the complexity of a software product by splitting the entire project in several smaller and better manageable pieces of software. This means, complex systems can offer great opportunities for customers, but complexity also represents one of the best arguments for security investigation:

> "The future of digital systems is complexity, and complexity is the worst enemy of security."
> (Bruce Schneier)[2]

Hence, stated requirements for functionality within an e-learning system will have to be discussed with strong relations to security in the course of this thesis to cope with this "worst enemy".

2.7.2 Security Restraints

Absolute security is only possible by absolute control, if at all. Hence, all activities would have to be logged and controlled in real-time to not enable fraudulent use or at least detect and stop it as soon as possible.

[2]Crypto-Gram Newsletter, March 15, 2000: `http://www.schneier.com/crypto-gram-0003.html` [03-11-2009]

However, despite the stated necessity for exhaustive security deliberations, we have to respect the secondary value of security measures [Wes08] and take care of the primary goal of e-learning: support the learning process. This means, supervision could be interpreted negatively, and therefore, it can lead to rejection. Privacy is considered very valuable, i.e., absolute control interferes with privacy and autonomy demands – especially of adult learners [KHS05]. Thus, security cannot be implemented using all possible logging mechanisms; otherwise, learners could be repelled. An important step to do is communicating planned supervision and mentoring intentions. Transparency is required to inform learners about the organisational framework including aspects of how the learning progress will be observed and where private communication is possible. In addition, especially for communication facilities, it is desirable to announce who is currently listening to communication, e.g., in chat rooms, and what activity can be considered to be private or teacher controlled.

Furthermore, all tasks besides learning, e.g., technical administration or interacting with security mechanisms, should be comprehensive and be reduced to a minimum for not distracting more than absolutely necessary. Security demands and implemented means must not be so time-consuming and laborious that they become primary. On the one hand, this concerns technical reliability aspects, e.g., reliable data storage, version control, and concurrency management in collaborative learning. On the other hand, security relevant interaction like authentication and authorisation should be kept simple to use.

The conflict of security aim vs. privacy and reduced distraction results in requirements for appropriate privilege management with transparent announcements of possibly observable activities. Privileges have to be assigned, such that private elements are possible to enable personal generic communication, as well as supervised learning phases with tutor access. These areas have to be strictly separated. Privacy and demands for self-direction must be accepted as far as possible, and as such, security demands need to be lowered to a certain extent in favour of more comfort for learners.

> "Security features usually increase the cost of a system and may make it harder to use. Before designing a secure system, therefore, one should identify the specific threats against which protection is required." [ITU91, p. 35]

Hence, the threat analysis in this thesis is meant to actually investigate the threat situation for e-learning and to provide decisions about the required security level. We need to realise that missing acceptance of (and usage by) learners in the system is even worse than missing *perfect* security.

2.8 Conclusion

First, typical activities and common tasks, as well as specific requirements for e-learning were considered – without being limited to security in e-learning literature. Second, the informatics research fields of software engineering and security engineering were examined concerning criteria and common activities in web-based systems. Extracted tasks and activities to be implemented in e-learning systems are collected and illustrated in Tables 2.2, 2.3, and 2.4. For referencing collected activities in subsequent chapters, the terminology as introduced in Section 2.4 will be used, i.e., the place holder symbol "□" as depicted in the tables will be replaced by respective indices.

Especially when regarding the amount of different activities for fulfilling demands from educational science, a certain degree of complexity in e-learning systems and a corresponding demand for flexibility in its construction get obvious. Note that practical e-learning systems do not necessarily implement all mentioned activities, but only a specific subset. Nevertheless, for a general threat analysis, all mentioned activities will be considered in the following, such that for a con-

Table 2.2: Activities in e-learning systems from an educational science perspective

	$\mathcal{A}_{\square,1,\square}$: learning process	$\mathcal{A}_{\square,2,\square}$: learning support	$\mathcal{A}_{\square,3,\square}$: learning management
$\mathcal{A}_{1,\square,\square}$	1. generically communicate 2. participate in intended cooperation 3. mutually assist each other 4. manage groups (as teacher)	1. mentor learners in case of problems 2. moderate cooperation	1. join group (as learner)
$\mathcal{A}_{2,\square,\square}$	1. interact with influencing learning contents 2. interact with systems providing feedback	1. evaluate chosen learning paths	1. create learning paths
$\mathcal{A}_{3,\square,\square}$		1. maintain (context-sensitive) online help 2. educate system related aspects 3. provide hotline support 4. track and evaluate user activities 5. conduct evaluation survey	1. maintain FAQ 2. maintain sitemap
$\mathcal{A}_{4,\square,\square}$		1. participate in spontaneous cooperation 2. include illustrative dynamic resources	1. (re-)engineer courses 2. use different action forms 3. apply different social forms
$\mathcal{A}_{5,\square,\square}$	1. interact without changing learning contents 2. manage bookmarks 3. manage personal notes 4. manage annotations	1. maintain user profile and portrait	1. manage course participation

Table 2.3: Activities in e-learning systems from a software engineering perspective

$\mathcal{B}_{1,\square}$: usability	$\mathcal{B}_{2,\square}$: functionality	$\mathcal{B}_{3,\square}$: architecture
1. manage designs 2. personalise appearance 3. archive text messages	1. provide and enable proper functionality 2. manage connections to external services 3. register user account online	1. manage metadata 2. distribute and connect to services 3. extend system 4. manage modules

Table 2.4: Activities in e-learning systems from an information security perspective

$\mathcal{C}_{1,\square}$: human factor	$\mathcal{C}_{2,\square}$: program logic	$\mathcal{C}_{3,\square}$: infrastructure
1. configure system 2. state (informal) security policies 3. apply content protection 4. prevent and detect cheating	1. manage user accounts 2. authenticate users 3. deploy PKI 4. manage privileges 5. apply roles-based access control 6. clearly separate duties 7. state and enforce (formal) security policies 8. support privacy concepts	1. change client computers 2. ensure anonymous communication

crete e-learning system the respective set of threats can easily be extracted and transferred. With respect to the quantity of identified activities, from a software engineering perspective modularisation and connections among several independent software components is reasonable. These components can overtake single functionality to achieve flexibility and to improve reliability. In addition, concerning information security aspects, numerous tasks and activities were mentioned in literature that emphasise human interaction. Therefore, fallibility, obliviousness, or bad intent can be considered to be the root of many security incidents in e-learning. The "human factor", i.e., activities of users and human error, is particularly indicated to be an important facet of security in e-learning. Consequently, this has to be considered when creating appropriate recommendations for e-learning systems in Chapter 6. Security education and technical support in case of problems, thus, gain in importance.

In summary, numerous scenarios and related activities are conceivable that might threaten affiliated data and users, or that could influence other activities and result in distraction of learners. Consequently, mentioned activities in the following chapter will be used to extract (technically manageable) assets that need to be protected in e-learning. The direct deduction of relevant assets from activities is sensible to ensure that the systematic threat analysis as described in Chapter 4 can specifically be conducted for the domain of e-learning without excessive consideration of general software-based or hardware-based threats.

Chapter 3

Preparation of the Threat Analysis

3.1 Overview

Security is a relative term closely related to the actual threat situation. Approaches aiming at an all-in-one solution for a general, highly secure, and still practically applicable security concept usually tend to fail [Sei06]. To avoid such problems, the threat analysis as described in this thesis will base on e-learning specific concepts and processes. Figure 3.1 sketches the current step of identifying assets within the whole analysis process:

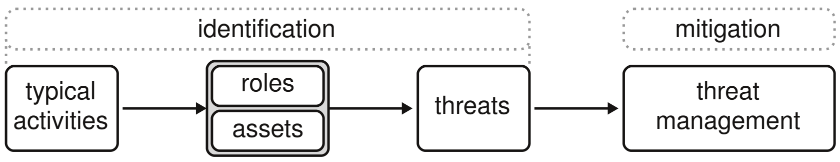


Figure 3.1: Current position in analysis process – step 2

Starting with findings and results of Chapter 2, assets and related roles for e-learning will be deduced in the following. In this context, assets mostly represent valuable data elements or concepts that are worth protecting in the domain of e-learning. These assets per se are independent of the actually applied methodology for the threat analysis, since they only describe valuable elements of the considered target systems. Hence, this step of identifying assets is independent of the subsequent step of analysing threats.

Bear in mind that assets most probably will represent data of different kind. Obviously, from an organisational perspective, assets have to be treated differently with respect to their actual value within the examined domain, e.g., learning contents are different to private communication data. Furthermore, data elements can be of different scope and value concerning legal aspects, e.g., data related to personal identity explicitly are covered by data protection legislation while other data elements might not be protected by any form of regimentation. But despite such differences in an organisational layer, from a technical perspective almost all of these data can be considered uniformly as entries in database relations or other storage systems without any difference in their semantics and relevance for the actual application. This especially implies that attacks against corresponding storage systems will affect all data assets at once. Nevertheless, for the sake of focusing on domain specific threats, in this chapter we will consider assets from an organisational perspective with partly neglecting actual technical data organisation and storage systems for this moment. However, technical threats still will be regarded within the threat analysis phase.

3.2 Role Based Access Control

Since there seems to be general agreement to role-based access control schemes for e-learning (cf. [Sch03b, KR04, Wei05]), in a first step we will take a closer look on required or conceivable roles for e-learning. This also includes an examination of a possible minimum amount of roles to not introduce interest conflicts reasoned by different duties and excessive privileges assigned to a single physical person.

Starting with an identification of required roles, several suggestions can be found in literature. In the IEEE Learning Technology Systems Architecture (LTSA) [IEE01] four entities are introduced that can be transferred to roles: learner entity, coach, evaluation, and delivery. While learner entity and coach can trivially be assigned to roles like learners and teachers, evaluation and delivery can be considered as (independent) third parties for conducting surveys and for the creation of learning material respectively. Weippl [Wei05] provides a similar division of roles. He distinguishes teachers, authors, learners, and managers. In this distinction, teachers and learners represent the main roles involved in the learning process, authors are responsible for the creation and design of learning material, and, finally, managers represent executives of an institution who are responsible for the whole system, but are not directly involved in the learning process in any way. Furthermore, Kerres proposes several persons for the conceptualisation of didactic media [Ker01, p. 171], since in his opinion it is generally not possible to let this be done by a single person – neither by teachers nor authors. This especially includes persons with topic-related expertise, persons with didactic design competence, and persons with topic-related didactic experience. According to Kerres, it also depends on the quality of this cooperation, whether the resulting media concept will be adequate in content and didactics. Similarly, Bruns and Gajewski [BG02] divide didactically experienced persons from those who actually implement the contents. However, the most interesting approach of Bruns and Gajewski is the consideration of roles as linked to different types of e-learning systems. In their opinion no uniform separation of roles does exist, since everything depends on the regarded e-learning scenario (cf. Fig. 3.2). Besides such variable sets of roles, some roles are introduced as omnipresent, e.g., administrator for building and maintaining the technical infrastructure and a responsible project leader for managing the e-learning project as a whole. Despite their general existence, this does not necessarily require these roles to be still present at each institution after deployment of an e-learning system. For example, the role of a project leader is to be in touch with persons not directly involved in the learning process, e.g., representatives, public relation personnel, administrators of formal concerns, technicians, or legal advisers. An administrator in this context is responsible for the technical aspects like an appropriate infrastructure. Thus, technicians/administrators – possibly only as consignees of the project leader – build up an infrastructure that is capable of planned tasks. A further omnipresent role, although not explicitly mentioned as such in [BG02], is given by learners. This role especially suggests a distinction of roles in global and local scope. This means, all users in the system independent of their global role can be locally in the role of learners within a course of some (other) teacher.

Starting with examples as given by Bruns and Gajewski [BG02], e-learning types roughly can be divided into five variants as illustrated in Figure 3.2. Note that besides the pure presence of roles, this figure also introduces the concept of internal and external roles. In this context, “internal” refers to employees of the same institution that actually use the e-learning system, while “external” means to pay third parties for doing the corresponding job. With respect to differences in those e-learning types, the first type is considered as product that was purchased by an institution to educate own members using small pieces of information. These e-learning contents must not be too large in size, since there are no tutors present to support in case of problems. Type 2 also considers e-learning content to be produced by external authors, but tutor support is also available as a service purchased for the institute’s members. For types 3 and 4, the concepts of purchasable products and services were partly or entirely incorporated by a

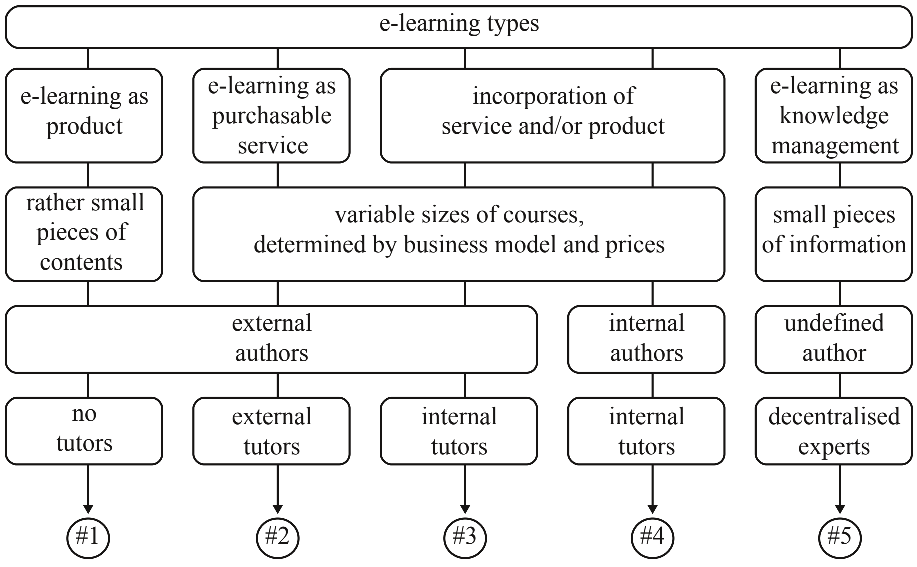

e-learning types
e-learning as product
e-learning as purchasable service
incorporation of service and/or product
e-learning as knowledge management
rather small pieces of contents
variable sizes of courses, determined by business model and prices
small pieces of information
external authors
internal authors
undefined author
no tutors
external tutors
internal tutors
internal tutors
decentralised experts
#1
#2
#3
#4
#5

Figure 3.2: Different types of e-learning scenarios

regarded institution. Thus, either tutors were educated and designated to support other learners within the own institution or, for type 4, even the contents were produced by internal authors. Note that the size of courses in types 2 to 4 depends significantly on the business model and the amount of purchased services. A special type of e-learning is given by knowledge management (type 5; cf. [BG02]), since there is a central system that can be used to store instructions and procedure descriptions by several employees to support others. In addition, instead of centrally organised tutors, contact information to specialists within the same company can be given for further support in case of problems.

In the following, assets will be directly deduced from activities as outlined in Chapter 2. Roles will be involved to support the discussion. This primarily means, that respective viewpoints of different roles have to be considered to take care of possible conflicts in expectations connected to assets.

3.3 Identification of Assets

In Chapter 2 tasks and activities were introduced and classified using a strict thematic division in disciplines and layers with respect to the proximity to users. In the following, former assignments will be refined by additional classes to provide more accurate classifications. Thus, activities will be reorganised and put in chronological order with respect to the learning process or an emulated attack sequence. The additional examination procedures aim at a thorough coverage of important aspects and a more meaningful identification of assets by combining all introduced activities to describe a generic e-learning scenario.

3.3.1 Procedure

Eibl and Schubert presented design criteria for e-learning and explicitly outlined connections between these criteria and dependencies as given for software engineering and information security [ES08a]. In the following, this approach will be refined with adapted sets of criteria for each discipline (cf. Tab. 3.1). Thus, design criteria for e-learning as presented in Section 2.5 are extended by the criteria of “equal opportunities” to emphasise the aspect of various types of learners and different technical and organisational conditions. For software engineering, a classification as given by the standard ISO/IEC 9126-1 [ISO01] will be used. As a third set of criteria, information

Table 3.1: Sets of criteria

e-learning		software engineering		information security	
A_1	social support by cooperation and communication	B_1	functionality	C_1	authentication
A_2	activities of students as important step of the learning process	B_2	reliability	C_2	access control
A_3	priority to meet learning objectives	B_3	usability	C_3	data confidentiality
A_4	flexible learning	B_4	efficiency	C_4	data integrity
A_5	integration into learning environment	B_5	maintainability	C_5	non-repudiation
A_6	equal opportunities	B_6	portability	C_6	availability

security services as presented in the standard ISO 7498-2 [ISO89] are applied, extended by the commonly used criterion of “availability”.

These sets of criteria in the following are put into relation with each other to extract direct or indirect dependencies and to uncover common properties for building clusters for subsequent examination steps (cf. Fig. 3.3). Note that this step is necessary with respect to different characteristics of mentioned criteria. For example, reliability (B_2) and availability (C_6) although in different criteria sets have more similarities than efficiency (B_4) and maintainability (B_5). Hence, a simple transfer of columns in Table 3.1 to clusters for further investigation would be too vague and imprecise. Thus, starting with the social situation and the communication context of learners, the term “situated learning” [LW91] especially demands that communication capabilities that are implemented must be sufficient for sharing knowledge and for getting in contact with other learners in order to succeed. The social aspect and learning in groups explicitly will be considered in this thesis. Social support and cooperation (A_1) primarily depend on appropriate functionality (B_1) with subcriteria like accuracy and suitability. To support the security of respective functionality, data confidentiality (C_3) is required to keep private messages undisclosed to unauthorised persons. It should be left to the learners themselves to decide who is allowed to view private notes. In addition to this, cooperation and communication facilities require a sufficient level of reliability (B_2), which demands efficiency (B_4) and availability (C_6) for appropriately working with this functionality. An e-learning system in this context is called available if it is reachable over a network with sufficient resources every time it is needed. Hence, besides the pure existence of connectivity also efficiency is ensured. Reliability adds demands for business continuity and a minimisation of time periods for technical service. Failures, if any, may only lead to short interruptions. Data integrity (C_4) is required for ensuring data exchange with unaltered message contents. Modifications of transmitted or stored data must be detectable. If reasoned by technical defects, fault tolerance and error correction can be applied. If reasoned by fraudulent usage, originator and context should be revealed with the help of integrity assurance mechanisms. To provide data confidentiality (C_3) and data integrity (C_4), an appropriate access control (C_2) is required, which bases on a sophisticated authentication (C_1) of users. Obviously, it should be ensured that users cannot transfer or increase their assigned privileges even if they collaborate – at least not without the perpetrator being identified. Another aspect is given by learning-related activities. Since in constructivism learning is no longer considered as externally transfusing knowledge but as a constructive process in each learner, this implies learning to be considered as an active process. Activities of students as an important step of the learning process (A_2) also depend on a proper functionality (B_1). In addition, access control (C_2) is necessary for not negatively influencing other learners during learning activities. It is absolutely essential to guarantee that learning objectives are prioritised almost equivalently to presence teaching, i.e., distraction by organisational and technical issues is kept minimal. Priority to meet learning objectives (A_3) refers to a software design that takes care of proper usability (B_3) and protection of own data and results. Third parties should be prevented by access control mechanisms (C_2)

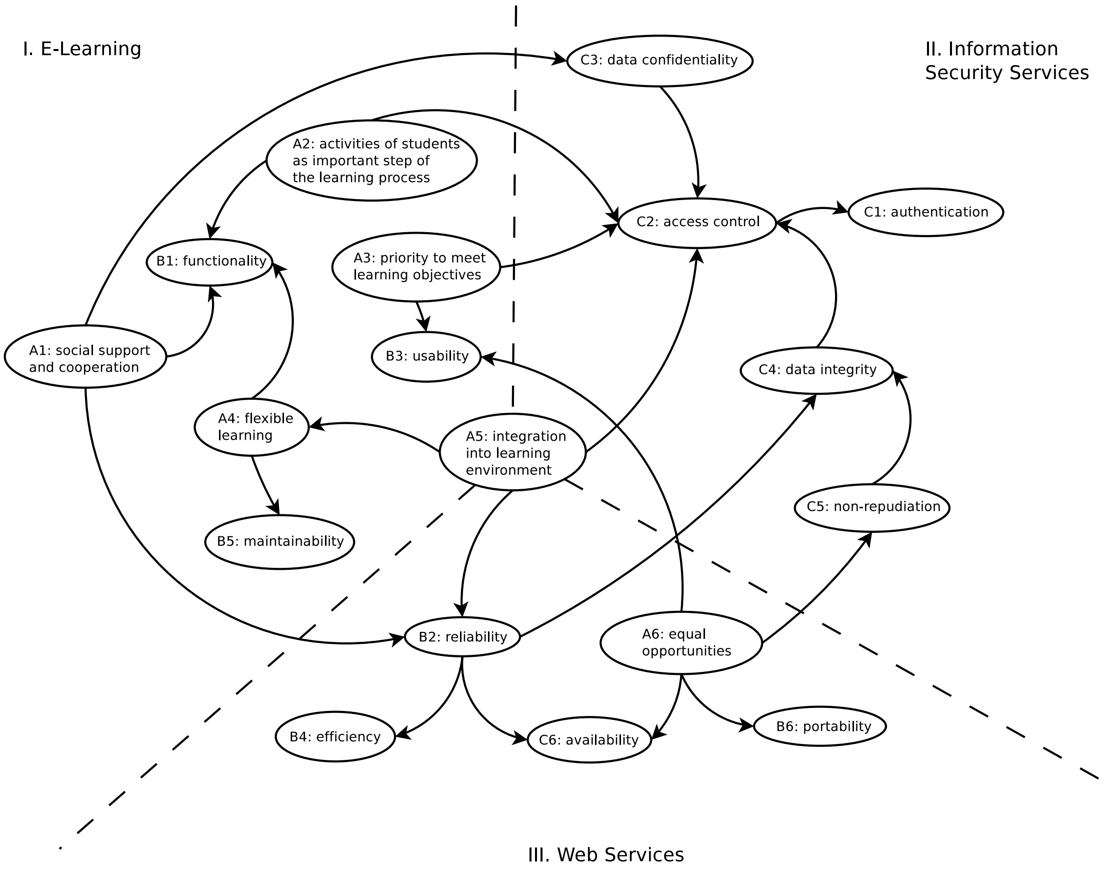


Figure 3.3: Dependencies among criteria

from changing foreign data on purpose or by mistake. Furthermore, each e-learning system has to be adapted to the special target group of learners including a learner-centred approach. Thus, flexible learning (A_4) describes possibilities to include and apply new activities fitting to required didactic deliberations with respect to the actual target group. Besides functionality (B_1), this also depends on proper maintainability (B_5) including modularisation and management of function modules by realising the subcriteria of changeability. Learning is a process which needs a carefully designed learning environment. The integration of e-learning components into the learning environment (A_5), therefore, depends on flexibility in learning (A_4) for not restricting teaching methods or didactic considerations, reliability (B_2) of the system to actually being able to use it for efficient learning whenever wanted, and access control mechanisms (C_2) to implement a separation of duties and roles as known from presence teaching. Finally, to reach many learners without increased restraints, equal opportunities (A_6) are aspired including demands for availability (C_6) and portability (B_6) referring to possible system-related limitations. In addition, equality is connected to usability (B_3) demands for not excluding less experienced users. With respect to mutual influence and possible denial of activities, non-repudiation (C_5) needs to be implemented for achieving equal opportunities in the sense of fair and equal treatment of users without possibilities for cheating on them. Note that this again requires data integrity (C_4) for not being undermined.

Having outlined dependencies and relations between stated criteria for e-learning, we can cluster them as sketched in Figure 3.3 by dashed lines. While the integration into the learning environment (A_5) affects all aspects and layers of e-learning systems, the other criteria can be assigned to one of three groups for identifying related assets:

I. e-learning (cf. Sec. 3.3.2),
II. information security services (cf. Sec. 3.3.3), and
III. web services (cf. Sec 3.3.4).

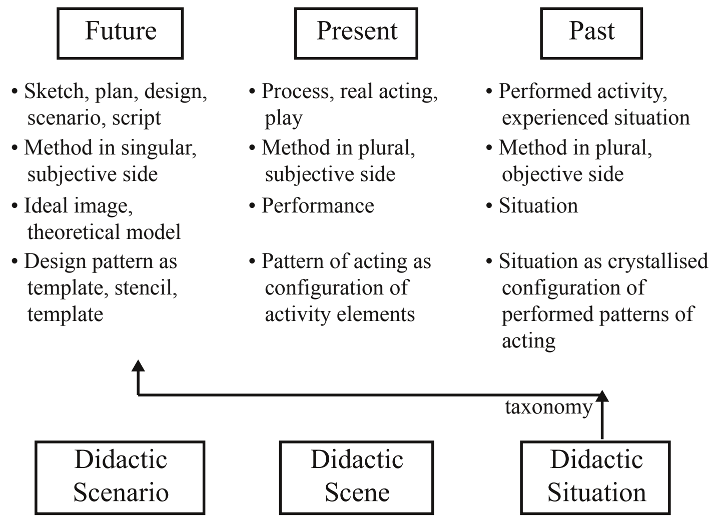

Future
Present
Past
• Sketch, plan, design, scenario, script
• Method in singular, subjective side
• Ideal image, theoretical model
• Design pattern as template, stencil, template
• Process, real acting, play
• Method in plural, subjective side
• Performance
• Pattern of acting as configuration of activity elements
• Performed activity, experienced situation
• Method in plural, objective side
• Situation
• Situation as crystallised configuration of performed patterns of acting
taxonomy
Didactic Scenario
Didactic Scene
Didactic Situation

Figure 3.4: Connection of Didactic Scenario, Scene, and Situation according to Baumgartner [Bau06, p. 61] (translated from German)

In the following paragraphs, the procedures for examining these classes will be discussed, before assets will actually be examined in corresponding sections.

Since "e-learning" consists of learning specific aspects, a structure following learning scenarios and didactic deliberations seems sensible. Hence, an approach similar to the taxonomy creation process of Baumgartner [Bau06] will be applied. Baumgartner uses concrete situations, i.e., performed activities and finished courses, as a starting point for the development of more abstract and freely applicable didactic scenarios (cf. Fig. 3.4). This development approach, on the one hand, aims at the creation of scenarios that can be applied to actual situations and, on the other hand, at the creation of appropriately usable taxonomy levels and methods. In the following, similar to Baumgartner we will introduce organisational and learning specific situations that can be used to deduce assets for a threat analysis. Furthermore, introduced situations will be brought into relation with the skeleton of a learning process. Phases used for this process base on a three-steps-model for organising classroom education in schools as presented by Meyer [Mey88, pp. 122-181]. In German, Meyer calls these steps "Unterrichtseinstieg"–"Erarbeitung"–"Ergebnissicherung". When translating this to English and adapting it from classroom education to e-learning, these steps can be referred to as "preparation of learning", "working on new content", and "repetition and assessment". Note that in this thesis for the last step only self-assessment will be considered (cf. Sec. 1.4.3). Besides the actual learning process, organisational steps prior to learning as well as postprocessing steps will have to be considered for covering all related situations. With the approach of, first, regarding preparatory steps, second, phases of the learning process, and, finally, postprocessing steps, this procedure represents a chronological structure from first contact with the e-learning system to a final cancellation of participation. However, possible cyclic repetitions within the process are not explicitly outlined in the following due to repeatedly affecting the same system components and assets.

For "information security services", no further transformation is required. Security services as introduced in [ISO89] already provide technical descriptions and are manageable by technical systems. Consequently, for an identification of assets related to security services, a procedure regarding criteria C_1 to C_5 with corresponding assets directly connected to these services will be followed. Criterion C_6 will be neglected in this group due to its closer relations to generic web services and dependencies on technical base systems rather than on organisational, system internal security implementations.

With respect to similarities of e-learning to software engineering projects in general, and to web applications in particular, for group "web services" a structure related to the multi-tier approach

Table 3.2: Integration of e-learning into the learning environment

E-Learning	Information Security Services	Web Services
⟨organisational steps prior to learning⟩	⟨authentication⟩	⟨network architecture⟩
⟨preparation of learning⟩	⟨access control⟩	⟨software components⟩
⟨working on new content⟩	⟨data confidentiality⟩	⟨data collection and exchange⟩
⟨repetition and assessment⟩	⟨data integrity⟩	
⟨postprocessing⟩	⟨non-repudiation⟩	

of distributing core services over a network seems sensible, e.g., putting the user interface management, program logic, and data retention on different physical systems. The distribution over a network can be very different according to the size of an institution and the concept behind the e-learning scenario. Interfaces and connected data, therefore, also have to be considered as well as reliability aspects and utilisation statistics. A structure resulting from this multi-tier approach can be reasoned as follows: "network architecture" for the actually implemented physical systems and the underlying network topology, "software components" representing software as distributed over the physical network, and "data collection and exchange" for all sets of data exchanged between these components and stored in corresponding software.

As stated in Section 3.2 assets in the following will be deduced from activities as outlined in Chapter 2, i.e., references to the $\mathcal{A}$–$\mathcal{C}$ entries as presented in Tables 2.2, 2.3, and 2.4 will be included where appropriate. In addition, identified assets will be highlighted in the text by giving their title in brackets. Those highlighted assets will be collected and presented in the form of a table at the end of this chapter. Table 3.2 provides a skeleton for the sake of illustration comprising introduced groups. Substructure elements for the investigation in subsequent sections are surrounded by angle brackets. These elements will be replaced in the final result table by appropriate assets.

3.3.2 E-Learning

This group concerns all activities related to actual teaching and learning processes. Thus, assets to be extracted in the following are related to learning and to the learners' progress or comprehension. Note that assets usually consist of several subitems that can be combined to more general ones for the sake of simplicity and manageability. Assets to be used for subsequent phases will be highlighted accordingly. As structure for this subsection, organisational and didactic situations in chronological order as presented in Table 3.3 will be used.

$\mathbf{O}_{1,1}$: registration

The amount of information to be stored during the registration process (cf. $\mathcal{B}_{2,3}$, $\mathcal{C}_{2,1}$) within an e-learning system can vary significantly according to the underlying business model and the aim of provided courses, e.g., issuing certificates. Typical data to be entered for a registration include personal information like (real) name, postal address, phone number, or email address. In addition, also personal system settings, desired courses, or certain options influencing a user's treatment within the system, e.g., a set of purchased services, may be stored. Once an account has been created, internal management information like a registration number or date of enrolment are stored, as well as a user portrait generated to provide other learners with shared information about this new user. This user portrait, of course, can, and should be managed by the user himself to share personal information on his own discretion (cf. $\mathcal{A}_{5,2,1}$). Concerning the identification and administration of learners, data directly related to learners are required – even if these data might

Table 3.3: Learning process and connected didactic and organisational situations

	phase	situation
org.	*organisational steps prior to learning*	$O_{1,1}$ registration $O_{1,2}$ seek and compare courses $O_{1,3}$ enrolment
didactic situations	preparation of learning (incl. motivation, back up starting knowledge)	$D_{1,1}$ clarification of organisational aspects $D_{1,2}$ test and include previous knowledge
	working on new content (incl. first acquirement of knowledge, consolidation, systematisation, application)	$D_{2,1}$ teaching basic information $D_{2,2}$ drawing connections to other topics $D_{2,3}$ application/transfer of knowledge
	repetition and assessment (incl. repeating, practising, self-assessment)	$D_{3,1}$ repetition of (whole) topic $D_{3,2}$ (self-)assessment
org.	*postprocessing*	$O_{2,1}$ evaluation of courses $O_{2,2}$ de-registration

not always be directly connected to the teaching and learning processes. Depending on internal structures of an e-learning system, administrative elements might be included in the core system. Thus, access to master data is necessary, e.g., for billing learners for participating in certain courses. Also, such data are needed for administrative purposes like contacting learners on a regular basis or notifying in case of certain events, e.g., the confirmation of course enrolments. With respect to data protection legislation, information of this kind explicitly should only be gathered for specific organisational purposes. No data may be collected only because of pure technical capabilities to do so. Note that due to the sensitivity of these data, even unauthorised disclosure of such information without further modification or exploitation, e.g., identity theft or social engineering, can lead to a huge negative impact on the institution's reputation. Processes in the system, like user tracking and logging all user requests (cf. $\mathcal{A}_{3,2,4}$), can possibly be exploited to extract patterns for identifying physical users behind recorded activities – even in case of anonymous data retention. Hence, personal data as affected by mentioned activities essentially need to be considered as an asset within an e-learning system ($\rightarrow E_1$: personal data).

$O_{1,2}$: seek and compare courses

Since courses and e-learning offerings can differ significantly, it is necessary for learners to seek and compare courses prior to enrolment (cf. $\mathcal{A}_{5,3,1}$) to assure choosing the best fitting course. This search includes a list of courses within a single e-learning system, or courses of different systems. Courses can be compared according to specific metadata information (cf. $\mathcal{B}_{3,1}$) describing their contents and organisational frameworks. For a learner, information about prerequisites, content descriptions, objectives, and level of difficulty are of interest to enable an appropriate decision concerning enrolment. With respect to prerequisites and larger learning modules, the integration of formal requirements for enrolment (cf. $\mathcal{A}_{5,3,1}$) can be included in the metadata to prevent learners from entering a course although not all relevant pre-courses were passed successfully. Assets related to this organisational step are given by course data like metadata and configurations concerning enrolment policies and possibly formal prerequisites ($\rightarrow E_2$: course data). Furthermore, formal requirements affect learning progress and results of former courses to provide formal conditions to be checked against before allowing learners to join a course ($\rightarrow$ E_3: learning progress and results). Finally, personal data are affected with respect to their status containing information about purchased services and included modules depending on the business model ($\rightarrow E_1$: personal data).

Note that the order of $O_{1,1}$ and $O_{1,2}$ can be changed according to the type of underlying e-learning system, i.e., in case of smaller e-learning modules to be purchased, seeking and comparing might take place prior to actually registering and signing a contract. On the other hand, for larger modules, e.g., a study at a distant university, registration takes place prior to seeking and comparing of provided courses for working towards the final exam.

$\mathrm{O}_{1,3}$: enrolment

After a learner has compared courses and a decision has been made, he can enrol in this course under certain conditions. While for $\mathrm{O}_{1,2}$ the payment for a specific amount or set of courses can influence search functionality, the purchasing status in this specific step can be relevant for providing access to a course. Furthermore, in case of limits concerning number of participants or further restrictions, negotiation processes might be necessary, i.e., learners cannot finalise their enrolment without administrative staff confirming this enrolment, and therefore, accepting this application for participation (cf. $\mathcal{A}_{5,3,1}$). On the other hand, even in courses free of charge, a mutual negotiation process might be implemented that prevents administrative staff or tutors of a course to manually enrol learners without their confirmation. This results in different approaches for enrolment, and consequently, different status information and data are required for managing actual enrolment procedures. With respect to purchased services, enrolment should be provided and billing mechanisms be integrated. In case of general services free of charge for a regarded set of learners, at least status information need to be considered, e.g., the fact of being enrolled. Access by affiliated persons to read or write such status information must be controlled, e.g., to prevent abuse by learners cancelling the names of other learners. Hence, assets affected by this step are the actual list of participants, the maximum number of participants for the specific course, formal requirements for being enrolled, and prerequisites to perform well within the course ($\rightarrow E_2$: course data) (cf. $\mathcal{A}_{4,3,1}$, $\mathcal{B}_{3,1}$). In addition, personal data are affected with respect to purchased services and business model for regarded courses ($\rightarrow E_1$: personal data). Learning results of former courses (cf. $\mathcal{A}_{2,1,2}$, $\mathcal{C}_{1,4}$) have to be considered within this step in case of formal requirements like successfully having passed another related course ($\rightarrow E_3$: learning progress and results).

$\mathrm{D}_{1,1}$: clarification of organisational aspects

Learners who have enrolled in a course in most cases only have information as provided by course metadata (cf. $\mathcal{B}_{3,1}$), i.e., they know the title of the course, content descriptions, prerequisites, and possibly learning objectives. But the more detailed organisational framework of what is expected from learners is not known entirely in advance. Thus, this first step within the actual learning process is concerned with initial preparation and accommodation to the topic. Courses in e-learning systems can be adjusted to specific didactic deliberations like organisation forms (cf. $\mathcal{A}_{4,3,3}$) or activity types (cf. $\mathcal{A}_{4,3,2}$). Thus, on the one hand, the organisation of a course and related demands for participants have to be outlined and communicated to them. On the other hand, the structure of the entire course has to be adapted accordingly to achieve appropriate switching points for changing organisation forms with not breaking the underlying learning paths as constructed for the contents (cf. $\mathcal{A}_{2,3,1}$). With respect to the structure of a course (cf. $\mathcal{A}_{4,3,1}$) and the increased difficulty to orient oneself in hypertext structures, course data also incorporate preparative and supportive activities. This includes documentations, supportive manuals, personal support (cf. $\mathcal{A}_{3,2,3}$), and separate courses about how to work with the system (cf. $\mathcal{A}_{3,2,2}$). Furthermore, orientation-related system components like a sitemap (cf. $\mathcal{A}_{3,3,2}$), public course metadata (cf. $\mathcal{B}_{3,1}$), and informative online help systems (cf. $\mathcal{A}_{3,2,1}$, $\mathcal{A}_{3,3,1}$) can be integrated. These aspects affect data concerning course organisation and metadata ($\rightarrow E_2$: course data), as well as initial learning contents ($\rightarrow E_4$: learning contents) to motivate further steps within the course and to reason the organisational planning.

$\mathrm{D}_{1,2}$: test and include previous knowledge

Even if formal prerequisites are fulfilled, informal requirements like previous knowledge cannot be guaranteed entirely. Thus, despite stated requirements, different previous knowledge has to be assumed which mostly relates to diffuse former experiences of learners. In this phase of the learning process, therefore, a common ground has to be sought to start upon. Usually real life examples from everyday situations are supportive to motivate learners (cf. $\mathcal{A}_{4,2,2}$) and simplify their first contacts to new contents (cf. $\mathcal{A}_{5,1,1}$). With this approach, learners get an impression of the relevance of aspired objectives and might more easily accept to spend time on learning such matter. Exemplary applications of these contents can be helpful and support a top-down approach – even providing an overview of further relevant topics. This phase in its whole is closely related to the asset of learning contents. This includes an appropriate start for learners with different previous knowledge, corresponding application examples, and adequate introductions to provide an overview ($\rightarrow E_4$: learning contents).

$\mathrm{D}_{2,1}$: teaching basic information

After a clarification of former experiences and previous knowledge, new contents can be taught in this phase of the learning process. Although learning contents are probably the most intuitive kind of data in an e-learning system, further data are created in this step and have to be processed accordingly. Basic knowledge to be taught includes primary definitions, general concepts to build upon, or a superficial introduction to all related topics. During the users' activities within the system (cf. $\mathcal{A}_{2,1,1}$) data are automatically produced and stored by the system (cf. $\mathcal{A}_{3,2,4}$). Furthermore, user generated contents have to be managed. In many e-learning systems learners can write personal notes as reminders or to support their understanding of some contents (cf. $\mathcal{A}_{5,1,3}$), they can annotate and highlight learning contents similar to text-markers (cf. $\mathcal{A}_{5,1,4}$), and they can manage bookmarks to access relevant contents more quickly (cf. $\mathcal{A}_{5,1,2}$). Although basic information is mostly related to pure textual definitions of terminology and basic concepts, with respect to constructivist learning theories as discussed in Chapter 2, learning objects must be designed in a way that users can interact with contents to a great extent (cf. [Sch02, Sch05]). But high interactivity levels also lead to demands for reliable learning paths (cf. $\mathcal{A}_{2,3,1}$) to not confuse learners in their way through the content. Whether content in general should be made available to external users in addition to registered learners (cf. $\mathcal{C}_{2,2}$, $\mathcal{C}_{2,4}$), e.g., by providing a guest role, depends on the specific situation and may introduce or even increase further issues like copyright protection (cf. $\mathcal{C}_{1,3}$). Concluding these aspects, learners are in touch with several assets. These include the learning contents with all subelements like correctness of contents, interactive elements with the possibility of uploading data objects for influencing appearance (cf. $\mathcal{B}_{1,1}$, $\mathcal{B}_{1,2}$), and user generated contents ($\rightarrow E_4$: learning contents). Besides contents, the personal notes and data created during the learning process are of importance, e.g., annotations, bookmarks, notes, and reminders ($\rightarrow E_5$: personal notes).

$\mathrm{D}_{2,2}$: consolidation and connections to other topics

On top of basic information, further information and topic related themes can be set to introduce the actual topics to be taught in a course. For example, when assuming a course about assembly language programming, basics as presented in step $\mathrm{D}_{2,1}$ could comprise the computation of binary numbers including the complement arithmetic or processor architectures. Further connections then can be drawn in $\mathrm{D}_{2,2}$ to program structures, conditional expressions as known from other programming languages and their corresponding representation in assembly language, and so on. Since in this step we can assume that basics are already known by learners, further didactic methods can be applied to activate learners and let them work on the contents independently to a certain extent (cf. $\mathcal{A}_{2,1,1}$, $\mathcal{A}_{2,1,2}$). Note that in this phase the consolidation of formerly learnt concepts refers to simple applications of knowledge to consolidate respective data and procedures and to internalise their applicability. Tasks and assignments in this phase are closely connected to problems as used for the motivation of corresponding learning contents, but do not include a transfer of concepts to completely different areas as dealt with in phase $\mathrm{D}_{2,3}$. Collaboration phases (cf. $\mathcal{A}_{1,1,2}$, $\mathcal{A}_{4,2,1}$) and sharing of personal notes can be integrated in this phase to provide peer reviews, mutual support (cf. $\mathcal{A}_{1,1,3}$), or contest situations to encourage learners to work more efficiently. Besides peer support, of course, communication between learners and tutors need to be established for supporting learners during their learning progress (cf. $\mathcal{A}_{1,2,1}$) and for preventing discouragement in case of emerging situations learners cannot cope with on their own (cf. $\mathcal{A}_{2,2,1}$). For the sake of usability, exchanged messages should be archived and deleted only on a user's discretion (cf. $\mathcal{B}_{1,3}$) in order to provide easy access to former communication threads. Besides learning contents and personal notes as already mentioned for $\mathrm{D}_{2,1}$, therefore, collaboration and communication data are affected by this step. Note that communication and certain collaboration steps can take place in almost all phases of the learning process, but in this particular step, they can be explicitly made part of the learning path to achieve further insights. This includes not only data like discussions during collaboration, proposed solutions, or interim and final group results ($\rightarrow E_6$: collaboration data), but also generic communication (cf. $\mathcal{A}_{1,1,1}$), i.e., talking about private thoughts to maintain social contacts, or specific communication concerning learning related topics ($\rightarrow E_7$: communication data).

$\mathrm{D}_{2,3}$: application/transfer of knowledge

This phase aims at a transfer of earned concepts and procedures to new areas that are not processed during former phases. Thus, learners should be prepared to apply their knowledge to different situations and recognise the abstract concepts behind concrete learning matters for adapting the concept accordingly. For this, different didactic methods can be used as well as applications to enable learners to practically apply their knowledge, e.g., simulation systems, animations allowing to influence its process by entering parameters (cf. $\mathcal{A}_{2,1,1}$), exploratory systems with very high degree of freedom for interactions, or collaborative projects (cf. $\mathcal{A}_{1,1,4}$, $\mathcal{A}_{1,3,1}$) to elaborate adapted processes for specific new situations. While exploratory systems entirely are covered by interactive learning contents as described before in phases $\mathrm{D}_{2,1}$ and $\mathrm{D}_{2,2}$ including a variety of available learning paths to integrate a reasonable degree of freedom (cf. $\mathcal{A}_{2,2,1}$, $\mathcal{A}_{2,3,1}$), project work referring to collaborative systems can be considered in more detail. Collaboration consists of several interim results, e.g., early brainstorming sessions, first grouping of ideas, refinements of former results, and, finally, the result to be submitted and presented to the whole group of learners. This may or may not be moderated by a tutor to support learners in their cooperative learning process (cf. $\mathcal{A}_{1,2,2}$). In addition to this, almost the entire process of collaboration is accompanied by communication among learners within the same group or even communication between learners of different groups to share ideas and help each other with emerging problems (cf. $\mathcal{A}_{1,1,1}$). Thus, communication messages are also of importance independent of their contents with respect to mutual motivation or solution proposals. As assets for this phase, therefore, all elements related to collaboration ($\rightarrow E_6$: collaboration data) and communication data ($\rightarrow E_7$: communication data) can be mentioned.

$\mathrm{D}_{3,1}$: repetition of (whole) topic

Similar to $\mathrm{D}_{2,3}$ this phase aims at repeating and applying leaned concepts. In particular, this phase takes all relevant aspects during the course into consideration, and therefore, covers more various learning contents than the former phase. Typical realisations for e-learning include assignments covering the whole topic or animations including different views for allowing some learning matter to be seen from different perspectives (cf. $\mathcal{A}_{2,1,1}$, $\mathcal{A}_{2,1,2}$). Repetition primarily has to be done individually to detect one's own problems in understanding and gaps in knowledge. Thus, this phase mostly is connected to appropriate learning contents and course data with respect to an integration of repetition phases within the learning paths (cf. $\mathcal{A}_{2,3,1}$). Hence, affected assets are related to contents ($\rightarrow E_4$: learning contents) and structural elements concerning learning paths, and possibly peer reviews ($\rightarrow E_2$: course data). Note that reviews again introduce communication data as further kind of asset ($\rightarrow E_7$: communication data).

$\mathbf{D}_{3,2}$: (self-)assessment

Besides the informal repetition phase as described for $\mathrm{D}_{3,1}$ a more formal process of self-assessment is sensible. This refers to explicitly integrated questionnaires, multiple choice tests, or assignments with using interactive animations, e.g., to create structures that can be automatically evaluated like finite automatons. If such submissions of assignment results can be checked automatically by the e-learning system, then the system can provide learners with feedback to direct them to their own mistakes or presumable gaps in understanding (cf. $\mathcal{A}_{2,1,2}$). Furthermore, if results of self-assessment offerings and interactive animations are collected and stored within the system, then such data collection can be used to create a profile of each learner (cf. $\mathcal{A}_{3,2,4}$). Obviously, besides questions, also assignments ($\rightarrow E_4$: learning contents) and their integration into the course structure ($\rightarrow E_2$: course data), as well as results and connected information about the learning progress are reasonable assets to be considered ($\rightarrow E_3$: learning progress and results). This also includes learner profiles and histories of former results in comparable self-assessments.

$\mathbf{O}_{2,1}$: evaluation of courses

After a course has been finished, for the sake of quality management, impressions and views of learners about the organisation and contents of a course should be collected (cf. $\mathcal{A}_{3,2,5}$). This can easily be achieved using questionnaires. Since honest feedback is desired, such evaluation processes are meant to be conducted in anonymous way (cf. $\mathcal{C}_{2,8}$, $\mathcal{C}_{3,2}$). Thus, the identity of learners should be kept hidden to parties evaluating given responses. However, since no learner should be able to bias results by voting several times, the identity of learners still have to be verified and stored accordingly (cf. $\mathcal{C}_{2,2}$). This contradiction will have to be solved by appropriate technical means. To sum up, the secrecy of a learner's identity provides an asset in certain scenarios like evaluating courses ($\rightarrow I_1$: user identity).

$\mathbf{O}_{2,2}$: de-registration

Analogously to the registration process as described for $\mathrm{O}_{1,1}$, if no further course is wanted by the learner, or necessary for being issued a certificate, then the learner can de-register from the system. This process affects all data related to the corresponding user. Data that was created during the active participation of the user in the system should be deleted from the system or at least the user's identity should be scrambled. Note that despite the necessity of keeping certain data, e.g., results of assessments, these data can still be removed from the actual system and be kept only on external archive media. Obviously, this includes personal data as required for administrative processes ($\rightarrow E_1$: personal data), but also information about the learning progress ($\rightarrow E_3$: learning progress and results) and the learner's notes during the actual learning process ($\rightarrow E_5$: personal notes). Although communication and collaboration data might not be changeable concerning the assigned identity of the de-registering user, the corresponding user's data, e.g., as stored in his mailbox ($\rightarrow E_7$: communication data) or personal data directory ($\rightarrow$ E_6: collaboration data), must be removed.

3.3.3 Information Security Services

This second group of criteria and assets considers all relevant data directly connected to the security concept of an e-learning system, and therefore, a straightforward application of mentioned security services as structure of this section is sensible. According to Table 3.1 and Figure 3.3 for this subsection the information security services C_1 to C_5 remain to be discussed. C_6 will be considered in relation to web services in Section 3.3.4.

C_1: Authentication

According to privileges that are depending on specific roles in a system (cf. $\mathcal{C}_{2,5}$), the authentication process is crucial. Users must be identified accurately to provide them with their legitimate set of privileges (cf. $\mathcal{C}_{2,4}$). Due to the focus on web-based systems in this thesis, the lack of states in the HTTP protocol results in further requirements to store relevant data in so-called sessions. After successful authentication, a session is created on the server comprising the current course environment and an identifier (ID) of the currently processed course, particular privileges and duties in the system, as well as current role assignment. Such session data usually are stored on server-side while a client has to provide the corresponding session ID to identify himself as owner of this session. Besides authentication credentials, therefore, this session ID is valuable and worth protecting. In case of software relying only on stored states in sessions, the authentication process could be bypassed by stealing and abusing this ID. Thus, as assets directly related to the authentication service, all credentials required for an appropriate and successful identification and authentication have to be considered. This includes username, passphrase, function or cryptographic key for challenge-response mechanisms, or further information involved in this process ($\rightarrow$ I_2: authentication data). Furthermore, information identifying and securing online sessions need to be protected, since otherwise relevant information to exploit currently active user processes could be disclosed and the system behaviour might be influenced by unauthorised persons ($\rightarrow$ I_3: session data). Due to the relevance of the authentication process for further security measures, several assets are indirectly affected by this process: for the sake of accountability successful or failed authentication attempts are logged ($\rightarrow$ I_4: accountability data), privileges are assigned to the user after successful log-in ($\rightarrow$ I_5: privilege settings), and policies related to the applied user account (cf. $\mathcal{C}_{1,2}$, $\mathcal{C}_{2,7}$) are affected and should be enforced ($\rightarrow$ I_6: policies and manuals).

C_2: Access Control

As mentioned in Section 3.2, e-learning systems in most cases apply role-based access controls (cf. $\mathcal{C}_{2,5}$) due to the mostly intuitive assignment of major global tasks to respective roles (cf. $\mathcal{C}_{2,6}$). In addition, an appropriate set of privileges and management facilities are required (cf. $\mathcal{C}_{2,4}$). Note that besides privileges to be managed in e-learning systems, the settings of base systems also have to be considered, i.e., access controls for directories and files on operating system level, grant rules and an appropriate user management for a database system, and privileges of additional processes, e.g., auxiliary programs for input filters (cf. $\mathcal{C}_{1,1}$). However, with respect to learners' demands for flexibility in choosing different computers to learn wherever they want (cf. $\mathcal{C}_{3,1}$), possibilities to restrict access to certain services depending on the used physical clients are limited. On a less technical layer, (in)formal policies can be regarded as a special kind of additional privilege settings (cf. $\mathcal{C}_{1,2}$). Although users cannot be generally prevented from conducting certain activities due to the lack of technically manageable kind, executives of an institution can state policies to demand appropriate user actions in listed situations. In case of formal policies, certain behaviour besides the actual access to data sets can be restricted (cf. $\mathcal{C}_{2,7}$). Assets primarily connected to access control, therefore, are the set of privileges and respective settings ($\rightarrow$ I_5: privilege settings), as well as information concerning stated policies and descriptions on how to act in the system ($\rightarrow$ I_6: policies and manuals).

C_3: Data Confidentiality

Although most of the assets as presented in this chapter are affected by data confidentiality or integrity issues, this paragraph is not meant to summarise those assets again, but to consider technical processes to ensure data confidentiality, and therefore, identify assets that are directly connected to the administrative aspect of implementing data confidentiality mechanisms. Obviously, this mostly is related to access controls and the authentication process. Besides this, as a special case of confidentiality the anonymity of users can be considered (cf. $\mathcal{C}_{2,8}$, $\mathcal{C}_{3,2}$). Similar to user identification and authentication, the confidentiality of a user's identity, i.e., anonymity, also can be important in certain scenarios. Learners in anonymous scenarios can be more honest without any fear of being treated negatively (cf. $\mathcal{A}_{3,2,5}$). Furthermore, anonymity can be applied to offer more comfort for learners in the sense of not being under surveillance (cf. $\mathcal{C}_{2,8}$), e.g., in communication processes. Assets related to this security service are given by access control mechanisms ($\rightarrow I_5$: privilege settings) together with basic functionality like authentication ($\rightarrow$ I_2: authentication data). For the aspect of anonymity, user identity and related information, including certain patterns within the system that would allow tracing data back to a physical person can be considered as asset ($\rightarrow I_1$: user identity).

C_4: Data Integrity

Unauthorised manipulation of sensitive data may have serious impact on the learning process and, in case of system files, such manipulation even can break the whole e-learning system. Consequently, to ensure this crucial security service, again the authentication process and proper access controls are of relevance (cf. $\mathcal{C}_{2,4}$). But furthermore, this service also has effects on the accountability and authenticity of transmitted and processed data. For example, to ensure that activities within the system can be traced back to their originators, certain information concerning accountability has to be stored and must not be manipulated by any party (see the following paragraph). Thus, assets directly connected to this service are access control mechanisms ($\rightarrow I_5$: privilege settings) together with a proper authentication process ($\rightarrow I_2$: authentication data), and data concerning accountability and authenticity ($\rightarrow I_4$: accountability data), since without integrity protection those services cannot provide reliable results due to possible changes by third parties.

C_5: Non-repudiation

Besides an administrative necessity to identify users for assigning privileges (cf. $\mathcal{C}_{2,4}$), users are meant to be accountable for their activities within the system, i.e., there is a demand for accountability and an evident connection between actions within the e-learning system and physical users actually initiating and processing certain tasks. Such a mechanism is sensible to trace illegal activities or consequential mistakes back to their originators. Note that non-repudiation and accountability explicitly are meant in both directions between all affiliated parties. No single party should be in the position to deny conducted actions and to repudiate the receipt of certain information necessary within the learning process. For example, in case of courses with limitations concerning the number of participants, learners may be allowed to register for that course and are issued a receipt of successful enrolment. After this mutual data exchange, none of these parties may be able to deny a registration, i.e., the learner cannot claim to have never registered for the course, and, on the other hand, the responsible person for the course may not repudiate to have accepted this enrolment due to the sent confirmation. Thus, as asset for the non-repudiation security service all data related to logging activities (cf. $\mathcal{A}_{3,2,4}$) and connected cryptographic mechanisms (cf. $\mathcal{C}_{2,3}$) have to be considered ($\rightarrow I_4$: accountability data). This kind of data obviously must not be changed by any person within the system, since otherwise its validity would be broken. Bear in mind that unauthorised access to such information also could provide exhaustive information about conducted activities. Consequently, for the sake of confidentiality privilege settings, authentication data, and session data as well can be treated as assets for this service.

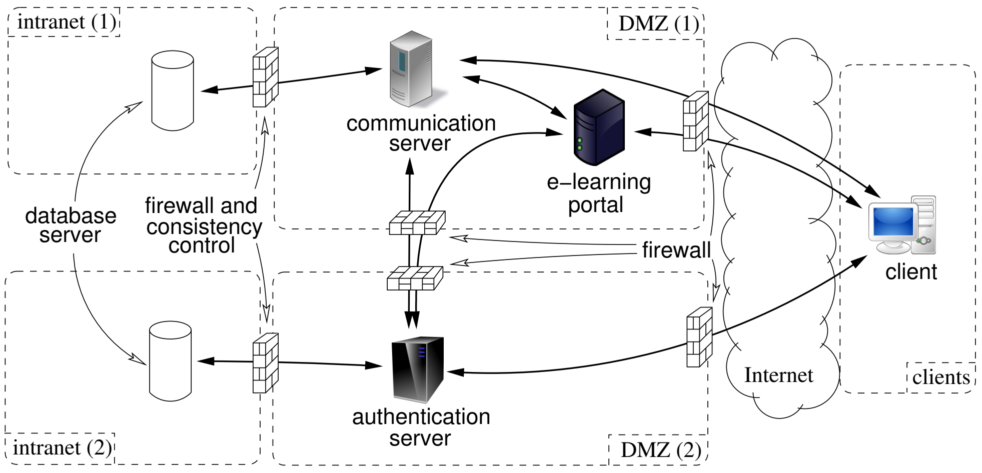


Figure 3.5: Conceivable architecture for e-learning with distributed components

3.3.4 Web Services

The investigation of e-learning in this thesis is limited to web-based systems, i.e., e-learning as web service inevitably is based on further technical components like web server or database system. Due to the client-server architecture of web-based systems, the infrastructure also includes the network architecture and respective interfaces of affiliated server systems for working together appropriately. Thus, besides the concrete implementation of the e-learning system itself, a consideration of the infrastructural basis including hardware, software, and administrative processes like adequate configuration is necessary to also identify valuable assets on a very low layer close to technical components, e.g., for protecting against abuse of network protocols. Furthermore, the continual shift to so-called modern web applications and Web 2.0 that imply the implementation of more and more functionality on client-side introduces additional attack vectors to be considered, e.g., client-side protection due to decentralised data processing for better flexibility and increased performance.

In this subsection a step-by-step attack procedure will be emulated to identify related assets that might provide likely targets for intruders and could be abused to gather sensitive information about the technical infrastructure and possible weaknesses.

Network Architecture

With respect to scalability and a separation of single services (cf. $\mathcal{B}_{3,2}$), the network architecture can differ significantly among various scenarios. Thus, an intruder, who only has a black box view on the network architecture, will have to do some examination of the topology and service interconnections prior to starting more specific attacks to single machines or software components. All services of an e-learning system could be placed on a single physical server or be distributed over the whole Internet. Since in distributed systems, e.g., as sketched in Figure 3.5, more than one physical system can be considered as potential target of attacks, the information where affiliated systems can be found is valuable. Complex network structures usually realise a separation of internal servers, demilitarised zones (DMZ) providing services for the Internet and intranet, and finally external services (cf. $\mathcal{B}_{2,2}$) placed on physical systems available over the Internet. Such networks can be analysed in certain borders and gathered information be used to choose the weakest target with most prospect of success when conducting an attack. Network topology aspects, furthermore, also affect flexibility for learners with respect to changing client machines (cf. $\mathcal{C}_{3,1}$). For the sake of reliability, required components might be available in redundant manner and possibly be hidden behind additional load balancing systems. In this case, reliability is understood in a technical sense, i.e., the system should be free of failures and outage including resistance against denial of service attacks to a certain extent. Thus, information about the network architecture and the interconnection of services must be considered valuable to prepare attacks ($\rightarrow W_1$: network topology). Reliability as generic asset is sensible to foster redundant architectures and an improved resource depletion resistance ($\rightarrow W_2$: reliability).

Software Components

In a second step, i.e., after the network topology could at least in parts be uncovered, attackers most likely aim at discovering software related details. Software most probably is never free of errors, and therefore, information about specific software details may reveal security issues that can be exploited. By providing information about the actual software product and its specific version, this information can be supporting for possible attackers when searching for already discovered and published software bugs. Hence, kind of software, specific product and vendor information, currently installed version, or operating system specific information should be treated as confidential and be protected accordingly. Network ports, protocols, or API specifications (application programming interface) can possibly be exploited, and therefore, belong to the information aspired by attackers. Activities in e-learning systems that are related to this second step of investigation foremost are the management of modules (cf. $\mathcal{B}_{3,3}$, $\mathcal{B}_{3,4}$) to provide further functionality (cf. $\mathcal{B}_{2,1}$) and the management of connections to external services (cf. $\mathcal{B}_{3,2}$), which results in ports to be opened for establishing network connections among services. From a software engineering perspective, a division into single services is crucial with respect to reusability and modularisation, but note that here particularly software components as functional modules are considered. Learning modules, e.g., distributed as SCORM packages, are not considered in this place, but will be regarded in subsequent chapters in the general form of archiving and restoring data sets of all kind. However, as concrete assets, several pieces of information about deployed software ($\rightarrow W_3$: software details) and its interfaces like the API or protocol specifications to be used for exchanging data with this software ($\rightarrow W_4$: interface details) need to be considered.

Data Collection and Exchange

Besides information about the presence of software and its connectivity, actual configuration details could provide an attacker with (conceptual) security issues, e.g., porous firewall rules. Related to this, unauthorised manipulations of configuration entries (cf. $\mathcal{C}_{1,1}$) even would provide possibilities to bypass other security mechanisms and to inject malicious code. In addition to software, hardware, and network details, information about the way how data are organised, e.g., the conceptual distribution of relations in a database including corresponding column names, are helpful for starting certain attacks. Besides database concepts, positions and names of configuration files, existence of specific software-related files, and files to be included by the main user interface can be used to re-engineer an application and collect hints about running services on the server even without full read access to these data. Attackers must be assumed to be quite smart in collecting and exploiting information that can simplify their work – even if this information seems irrelevant at first sight. Hence, unauthorised information disclosure should be prevented in general. Note that in case of improper server configurations, the amount of collectible data can be immense and the effort for collecting such data can be very low by simply utilising Internet search engines. Furthermore, a lot of data are produced while users are active in an e-learning system (cf. $\mathcal{A}_{3,2,4}$). Bear in mind that in case of security incidents attackers might gain access to these data, and consequently, activities of users including their possibly embarrassing vain endeavours for certain learning activities are completely revealed. In conclusion, with respect to the configuration of deployed systems and emerging data sets during productive use, we need to consider configuration data stored in files or databases as an asset ($\rightarrow W_5$: configuration details). Additionally, information about the way data are stored should be kept secret, such that attackers are prevented from discovering this structure to specifically prepare subsequent attacks ($\rightarrow$

W_6: data organisation). Furthermore, all sorts of events and statistically relevant activities have to be protected from unauthorised disclosure ($\rightarrow W_7$: log data and statistics). This latter aspect mostly is related to privacy issues (cf. $\mathcal{C}_{2,8}$) as emphasised to be of certain relevance to learners' acceptance (cf. Sec. 2.7).

3.4 Conclusion

The aim of this chapter was to prepare the threat analysis by identifying relevant assets for e-learning. For this, first, roles were discussed for a role-based access control mechanism with also incorporating the concept of subroles for auxiliary tasks to be delegated to additional persons. Second, sets of roles were brought into relation with types of e-learning systems, i.e., different types were discussed and considered concerning their amount of roles to assign all tasks without introducing conflicts. For the actual identification of assets, a division into the research fields e-learning, software engineering, and information security similar to the one of Chapter 2 was applied. However, while a thematic distinction into design criteria was chosen for e-learning specific aspects in Chapter 2, in this chapter a chronological sequence was followed. These different approaches aimed at a thorough and reliable coverage of e-learning specific activities and assets, such that the probability of neglecting important aspects could be minimised. For software engineering and information security, an extended set of criteria according to well-accepted standards was used to cover all relevant areas of the affiliated disciplines.

During the entire examination following sketched procedures, back-references to introduced activities of Chapter 2 were applied to ensure specificity for e-learning. Assets were directly deduced from tasks and activities as extracted from specific literature. A distinction in classes "e-learning", "information security services", and "web services" was applied to emphasise different characteristics of assets. Data and processes closely related to e-learning obviously differ from more general, but still relevant security-related assets like privilege settings, or general assets to be considered for all web services, e.g., reliability of the system. This distinction again will be picked up for the actual threat analysis to take care of organisational and more technically oriented threats. The resulting twenty assets are collected and presented in Table 3.4. For references to assets in subsequent chapters, the placeholder symbol "□" will be replaced by the corresponding number within table cells.

Table 3.4: Assets assigned to groups

$E_\square$: E-Learning	$I_\square$: Information Security Services	$W_\square$: Web Services
1. personal data	1. user identity	1. network topology
2. course data	2. authentication data	2. reliability
3. learning progress and results	3. session data	3. software details
4. learning contents	4. accountability data	4. interface details
5. personal notes	5. privilege settings	5. configuration details
6. collaboration data	6. policies and manuals	6. data organisation
7. communication data		7. log data and statistics

Each of the mentioned assets will be examined in Chapter 4 with respect to possible threats that might result in unwanted situations by harming or influencing corresponding data sets or states.

Chapter 4

Analysis of Threats in E-Learning Systems

4.1 Overview

The aim of this chapter is a systematic threat analysis basing on introduced assets of the former chapter (cf. Fig. 4.1). Thus, different threat analysis approaches will have to be compared and be discussed with respect to the stated demand of providing a systematic methodology.

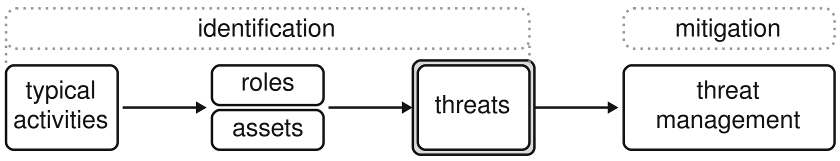


Figure 4.1: Current position in analysis process – step 3

Since threat analysis approaches usually are very technically oriented and are meant to be applied to technical systems that allow precise and accurate consideration of single component failures, we have to assume certain drawbacks when applying one of these approaches (cf. Sec. 4.2). Limits and possible mitigation of inherent drawbacks will be discussed for adapting the chosen methodology for analysing e-learning.

Assets will be considered with respective threats according to the group assignment as given in Chapter 3. Thus, learning specific assets, i.e., assets of group “e-learning” (cf. Tab. 3.4), will be discussed in Section 4.3. Note that the term “e-learning assets” in the following for the sake of simplicity will be used to refer to this asset group. This term does not imply that assets of other groups are not related to e-learning. However, the proximity to learning in this case is primary for this denomination. Assets of group “information security services” will be considered in Section 4.4 and assets of group “web services” in Section 4.5. The discussion of threats to assets will be aligned to different scenarios to provide more concrete examples and connections to practical e-learning systems. Respective exemplary scenarios for each group of assets will be presented at the beginning of corresponding sections. Related to this, the discussion of threats in the following does not cover the entire analysis results. Scenarios are only related to a specific subset of actually discovered threats within this research project, i.e., only an excerpt of these threats will be discussed in the following, while the complete set of results is given in the appendix of this thesis.

After a specific asset discussion, in Section 4.6, general conceptual weaknesses affecting several assets in the same manner will be discussed. Additionally, Section 4.7 provides technical threats with examples to emphasise common threats related to the technical basis. Finally, Section 4.8 will conclude results of this chapter and provide an outlook to subsequent chapters.

4.2 Threat Analysis Approach

Before actually conducting a threat analysis, the best fitting analysis methodology has to be determined.

> "While it is tempting to think that a security scanner or application firewall will either provide a multitude of defenses or identify a multitude of problems, in reality there are no silver bullets to the problem of insecure software. Application security assessment software, while useful as a first pass to find low-hanging fruit, is generally immature and ineffective at in-depth assessments and at providing adequate test coverage. Remember that security is a process, not a product." [MK07, p. 16]

Consequently, to identify threats not belonging to "low-hanging fruits" a formal analysis approach with structured methodologies is required that ensures better coverage of the considered object of analysis without increased risk of missing certain aspects. Furthermore, a visual organisation of actually relevant threats is reasonable for subsequent steps of mitigating discovered threats. Visualisation in this context supports keeping an overview of these threats and checking recommendations to mitigate problems against still open ones, which are not covered by security means or organisational frameworks.

4.2.1 Comparison of Approaches

Threat analysis methodologies can be roughly categorised into the following four classes (cf. [And01, Eck04]), where each of the classes may allow different variants as shown in brackets:

- state-based approaches (Petri nets, Markov analysis),
- table-based approaches (failure modes and effects analysis (FMEA), threat matrix [And01]),
- parameter-based approaches (hazards and operability approach (HAZOP)),
- tree-based approaches (fault tree analysis (FTA), attack tree [Sch99], threat tree [And01]).

With respect to stated requirements for the methodology, and specific characteristics of e-learning systems, all of these classes provide approaches with advantages and disadvantages for the actually aspired analysis in this thesis. While state-based approaches provide sophisticated models to construct and visualise mutual dependencies of single components of software, several simultaneous processes can hardly be represented. But especially with respect to web-based services like considered for e-learning (cf. Sec. 1.4.3), concurrent activities within the system can introduce security threats like race conditions, which would be entirely neglected when using such methodologies. Table-based approaches as second kind cannot be uniformly discussed due to the various character of respective approaches. While the threat matrix approach of Anderson bases on (non-systematic) collections of threats and possible mitigation techniques to be assigned to rows and columns, such that each threat should be met by at least one counter-measure, for the FMEA approach [DIN06] no concrete methodology for gathering possible threats is mentioned. Instead, the focus is put into probabilities of detecting and mitigating threats with also mentioning possible counter-measures. The HAZOP methodology as parameter-based approach does not specifically focus on threats that might occur through certain attack patterns or suboptimal organisational concepts, but instead available and known parameters in the system are considered with possible effects of misconfiguration or abuse of these parameters. Finally, tree-based approaches consider a root element as an unwanted event together with possible events leading to this event in nodes of the next level of the tree, such that each subtree is a recursive application of this analysis methodology. Different tree-based approaches are available that differ in their terminology, e.g., using different symbols to state Boolean dependencies of events in nodes of the same level. In addition, trees can be used to analyse the existence of threats, but also to analyse consequences of already appeared threats to uncover possible (side) effects of emerging problems.

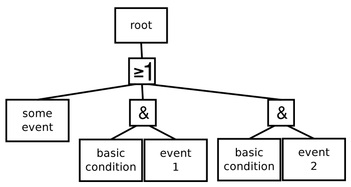
All of the mentioned threat analysis approaches are primarily developed for technical systems with very concrete subelements to be systematically analysed. Thus, none of the approaches can be directly transferred to an analysis of the area of security in e-learning without any drawbacks. However, drawbacks can be mitigated to a certain extent as discussed in the following subsection. With respect to mentioned advantages and disadvantages, most approaches clearly can be neglected for further discussions. Since web-based systems explicitly have to include concurrent access to certain pages, state-based approaches are not sensibly applicable and can be excluded. Table-based approaches lack a clear visualisation for getting an impression of mutual dependencies. In addition, they either are based on brainstorming sessions or focus on probabilities, which explicitly are neglected for the analysis in this thesis (cf. Sec. 1.4.3). Parameter-based approaches are reasonable in systems, where all variables are known in advance. Especially when considering extension modules to be loaded dynamically during the run-time of a system, this is no longer the case and the pure connection to possible threats depending on the existence of other factors needs to be in focus of the analysis. Hence, this approach also does not fit to the considered application field, so that only the tree-based approach is left to be considered in more detail. Different methodologies using the tree representation are available and used in certain situations with comparable effort and explanatory power. In the following, the fault tree analysis (FTA) methodology will be used. This methodology provides an intuitive tree-based approach with terminology similar to electrical engineering and a procedure that is standardised in DIN 25424 [DIN81].

4.2.2 Critical Discussion of FTA Approach

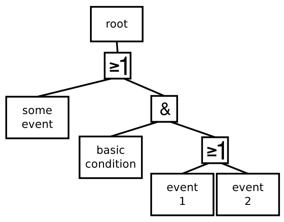
The design of the fault tree analysis is kept very simple, i.e., it only consists of events of different kind and logic gates to combine events and to include dependencies. Besides this simplicity which results in improved clarity of presentation, such fault trees convince by their structured approach for processing the analysis. The actual analysis for this considers the root element to be an undesired event that has to be prevented. Starting from the root so further subsequent elements can be integrated into the next level of the tree, where those newly integrated nodes represent undesired events that directly would lead to the undesired event of the root node. This process will be continued recursively for every uncovered interim event until basic events as leafs of the tree could be identified, which allow no further processing.

Bear in mind that the FTA methodology was initially meant for technical systems, where the termination can be specified easily by identifying an event directly related to a single component that cannot be considered in more detail, and therefore, a basic event in a leaf of the tree must have been reached. For e-learning such a termination condition cannot be stated in the same clarity, since besides technical failures also human failures must be considered. This can be done to a certain extent, e.g., organisational misconception like a lack of user education, but obviously it would exceed the scope of this thesis to analyse human errors in all facets including psychological background and personal situation. Hence, as termination condition for the FTA in this context reaching an event connected to human errors can be considered as well as reaching a technical component that allows no further consideration without losing focus on e-learning systems.

Disadvantageous in the FTA methodology is the inherent contradiction between logically reducible terms and a correct placement of directly dependent events in the tree presentation (cf. Fig. 4.2). This means, on the one hand, for a correct and supportive presentation, directly related



(a) redundant events with direct connection to dependent components



(b) logically reduced tree neglecting direct dependencies

Figure 4.2: Comparison of redundant tree and logically reduced representation

events should be placed close to each other and be combined by the appropriate logic gate, e.g., a logic AND gate for events that need to happen simultaneously for a superordinate event to actually happen as well. If some of these events are preliminary for numerous other events to happen, then such events, of course, have to be placed in redundant manner. On the other hand, logical reductions could be performed, i.e., the mentioned preliminary set of events also could be moved upwards in the tree and be reduced to only a single appearance in the (depicted) logical term with adapted logical structures. Both approaches have advantages and disadvantages with respect to complexity, redundancy, and clarity of dependency relations. Note that redundant subtree inclusion in parts can be prevented by using so-called transfer symbols as placeholder (cf. Sec. 4.3.1 and [DIN81]). Consequently, an immediate placement to the position where it acts as dependency can be ensured while only a single subtree representation would have to be maintained, such that error-proneness of redundancy can be reduced. Thus, in this thesis a logical reduction of the tree will only be conducted implicitly for the recommendations.

Another shortcoming of this standard is the fact that neither an inheritance concept for subtrees is included in the terminology nor a weighting concept for emerging events and their impact on the root element. Although this in parts is deliberate, it raises a problem concerning prioritisation of single events. Threats can significantly differ in their impact on the system as a whole, and therefore, providing a tree without any weighting of threats is only sensible, if all threats are aimed to be mitigated, instead of only focusing on most "dangerous" ones. In addition, specific events that already can be considered as threats to the system as a whole might also be exploited again by attackers in combination with other events to provoke even more damage. For example, the disclosure of personal data to unauthorised persons already can be regarded as security incident, but gathered knowledge again could be gainfully used to start social engineering attacks for even collecting more sensitive information (cf. [MS03]). Thus, from a logical point of view, in the fault tree the first mentioned negative event already results in a "true" value of the whole term, such that the additional exploitation of this event can only be considered as neutral element without any further effect – despite its practical difference concerning the impact of these events. However, for the sake of completeness and improved support for administrators, such exploitation subtrees also will be included in the fault tree, although their presence is irrelevant for the logical structure of the tree.

With these adaptations and decisions for consistently using the FTA methodology, some of the introduced shortcomings can be reduced. To avoid the problem of depending on the accuracy of own brainstorming sessions, and therefore, to increase reliability for a systematic methodology, catalogues of attack patterns and common weakness listings will be applied and inserted into the tree. As starting point, available security catalogues and penetration testing manuals will be used, i.e., [Swa01, SGF02, Gro03, BSI05, SH06, Her06, MK07, MBPC09, MIT10]. Threats discussed in these catalogues will be filtered, put into relation, and be appropriately extended to build up the fault tree as resulting from this e-learning specific analysis. Due to the size of the resulting fault tree, a direct embedding in the text of subsequent sections seems not sensible, such that only excerpts will be illustrated where appropriate while the entire fault tree is placed in Appendix A.

4.3 Analysis of E-Learning Assets

Actual analysis results will be presented and discussed using exemplary e-learning scenarios to emphasise differences in e-learning types as introduced in Section 3.2. But before going into details, in the following subsection an excerpt of the entire fault tree will be presented to introduce the terminology and to draw relations to assets as introduced in Chapter 3.

4.3.1 Introduction to Fault Tree Terminology

Besides presenting threats to e-learning assets (cf. Sec. 3.3.2), this section will be used to introduce FTA terminology and related symbols. In Figure 4.3 a fault tree is given representing the subtree dealing with e-learning specific assets. As example, the asset of personal notes in this tree illustrates a full path in the tree from the (relative) root, i.e., node representing an undesired event in the asset group "e-learning", to basic events in leafs of the tree.

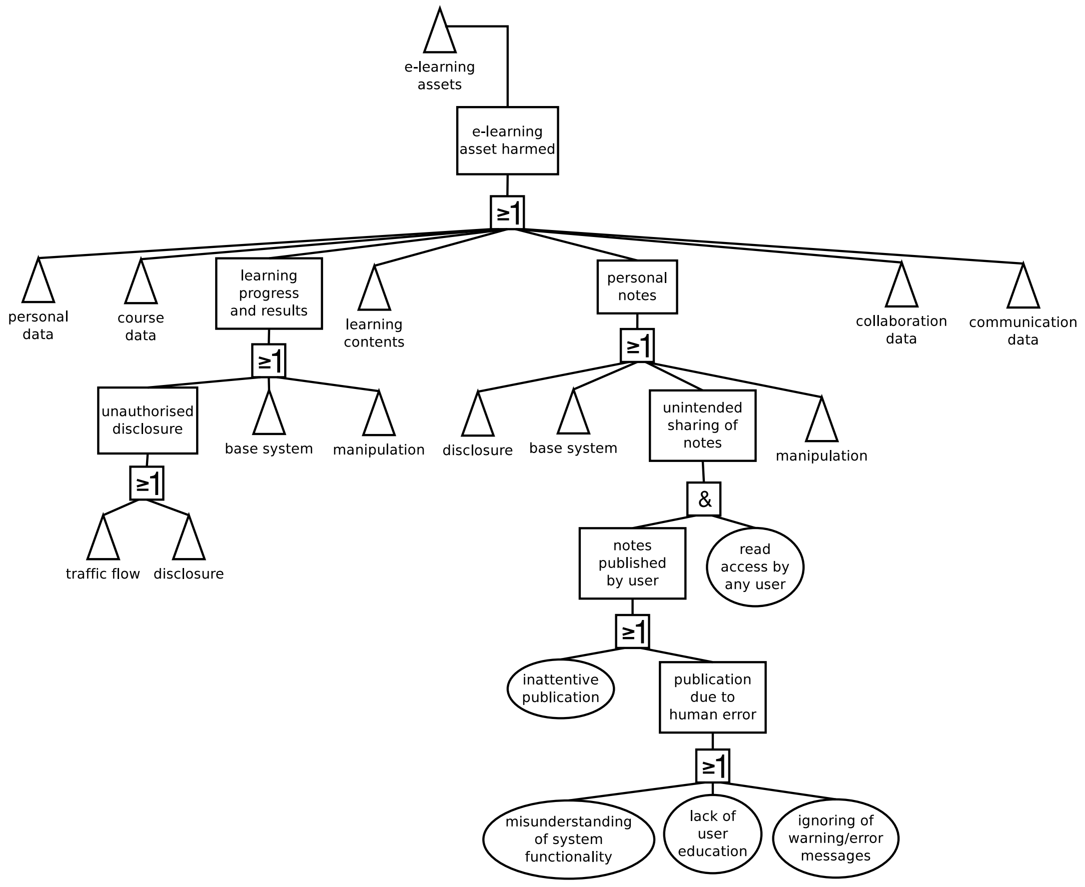


Figure 4.3: Fault tree excerpt for e-learning assets

The upper-most transfer symbol, i.e., triangular symbol named "e-learning assets" represents a connecting node to the main tree including the global root. Transfer symbols in DIN 25424 [DIN81] are divided in "transfer in" and "transfer out", where this mentioned upper-most transfer symbol means "transfer out", because of the logic flow to go upwards from leafs to the global root. As "transfer in" in the main tree an equally named transfer symbol has to be placed to correspond to this instance. The remaining transfer symbols in Figure 4.3, therefore, represent the "transfer in" type as placeholder for corresponding subtrees providing a set of threats that influence listed events. Note that transfer symbols not necessarily are meant as 1-to-1 relation, but rather n-to-1 with several "transfer in" nodes connecting to a single subtree instance linked with a “transfer out” symbol respectively. As further components of the tree, rectangular symbols represent interim events to be further analysed in the subtrees beneath these nodes. Basic events in leafs are represented by ovals. Logic gates are denominated as commonly used in electrical engineering, i.e., square symbols with ampersand character “&” for AND gate and “≥ 1” for OR gate. Note that the symbols for binary logic are not the ones originally contained in DIN 25424, but follow the revised and standardised version of IEC 60617, part 12 [IEC01].

For the actual procedure of analysing threats to e-learning assets, this means that the upper-most interim event “e-learning asset harmed” acts as relative root of this subtree, i.e., as unwanted event that will happen if one of the assets of this group is harmed in any way. Combined by an OR gate, for each of the asset nodes now the process continues, such that each node in the next level of the tree is considered as new relative root node for recursively uncovering further unwanted events. The recursion terminates in case of reaching a basic event (cf. termination criteria in Sec. 4.2.2).

4.3.2 Scenario: Purchased Advanced Training with Mentoring

For discussing e-learning assets, a scenario of a company will be used as reference, where the company members are provided with an advanced training to learn relevant matter for better coping with company related issues and for improving understanding of important aspects of the company’s value on the market. This e-learning course is purchased from an external provider who has implemented respective learning contents and offers support for learners within the company (cf. type 2, Fig. 3.2). Every participant of this course can be awarded with a certificate for successful participation, such that the provider of this course is obliged to include some assessment elements. This is realised by an assignment to be processes collaboratively by small groups and be submitted to the provider. The course has a specific start and end date. Applications for participation are collected by executives of the company. Issued certificates also will be sent to executives and be distributed by them to honour respective company members. Affiliated roles in this scenario are learners as participants of the course, tutors for supporting and moderating collaborative processes, course creators within the provider’s company, administrators to set up the e-learning system, and executives of the own company to promote and supervise this kind of training. Since this training is meant on a voluntary basis and not obligatory for all company members, the staff council has demanded that the actual learning progress and personal data have to be protected by the system for the sake of privacy protection and avoidance of negative consequences for participants. Thus, no unauthorised person may extract such data from the system. This especially holds for executives of the company who may only know who is participating, but not how they perform within the learning process. Furthermore, also mentors or other learners should not be in the position to gather exhaustive information about participants except for relevant information to work with them on collaborative assignments. Administrators are meant to be the only ones to access the profile of learners and to manage administrative elements like master data to a required extent for managing participation.

4.3.3 Threats of E-Learning Assets

Basing on the given scenario, threats to e-learning assets will be investigated in the following. The assets in their enumeration as given in Table 3.4 will be used as structure of this subsection.

E_1: Personal Data

Starting with the first asset of personal data within the system, we have to consider the actual necessity of collecting such information. However, with respect to administrative processes like managing participation, contacting learners, or issuing a certificate, master data are required. The collection of master data for e-learning can be considered as explicit data collection with users being informed about it. Such master data usually contain common contact information like e-mail addresses, postal addresses, phone numbers, personal information like day of birth, nationality, or native language, and administrative data like internal learner ID, list of purchased modules and courses, special conditions assigned with this learners, or results of completed courses. A sharing of specific pieces of data, e.g., contact information, with other users in the system is sensible. But, obviously, due to the sensitivity of such data, universal access for all users must be avoided. In the given scenario this means, that administrators, of course, should be able to access and manage such data, but tutors may only access contact data for directly communicating with learners, e.g., via their e-mail addresses or similar. Implicit data collection, on the other hand, mostly consists of log files that store exhaustive information about user activities. This kind of data usually has no need to be accessed directly by users of the system. Nevertheless, logging mechanisms are reasonable in this scenario to support non-repudiation by recording critical activities in the system, e.g., the submission of assignment results.

Threats related to explicit data collections are given by erroneous privilege settings or by inattentive publications, i.e., users share contact information with others by mistake. Note that master data even could be disclosed indirectly. For example, search mechanisms, which internally access master data, might reveal sensitive information, although the exact content still is kept hidden in the front-end. Hence, instead of a clear disclosure of sensitive information at least relations to other data sets can be discovered and the existence of entered strings is confirmed. By a stepwise refinement of search strings, exact information can be extracted from the database system. For implicit data collection without appropriate access controls, the fact of having excessive log entries with missing anonymisation could provide an attacker with valuable information about personal data of learners, their interrelations, and learning progress. Even with anonymisation, patterns as given by contemporary events or access to related elements can provide information about traces of single persons in the system.

General threats are given by low level data manipulation and disclosure as described in Section 4.7 and circumvention of system internal privilege settings by abusing import/export mechanisms as further discussed in Section 4.6.3. Furthermore, note that not only the discovery of single pieces of data, but also the reuse of such information to gather even more information can be useful. This can support identity theft by combining single pieces of information to pretend being someone else. Bear in mind that especially when aiming at deceiving users, every piece of information that is inauspiciously distributed by the e-learning system can be exploited to support the act of deception (cf. [MS03]).

E_2: Course Data

Course data include all information directly related to courses, e.g., course settings, list of participants, and metadata descriptions. This asset mostly is not directly threatened by asset specific problems, but rather by general conceptual problems or concept independent low level attacks.

In the given scenario, course settings include important dates, e.g., start and end date of the course or deadlines for assignment submissions. Thus, manipulations might aim at achieving more time for preparing solutions to collaborative assignments. Threats related to this are mostly basing on access control mechanisms that are not tight enough to protect against unauthorised access. Also general disclosure and manipulation by exploiting low level weaknesses are possible (cf. Sec. 4.7). Concerning manipulation of participant lists, several system internal mechanisms might be exploitable related to improper privilege settings and missing separation of duties (cf. Sec. 4.6.1). This means, if tutors who are only meant to support learners in the sketched scenario can enrol or delete learners without their confirmation, then newly enrolled learners can be subordinated to tutors who usually are equipped with more privileges than plain learners. Thus, even other assets like personal data could be affected by this threat. On the other hand, if

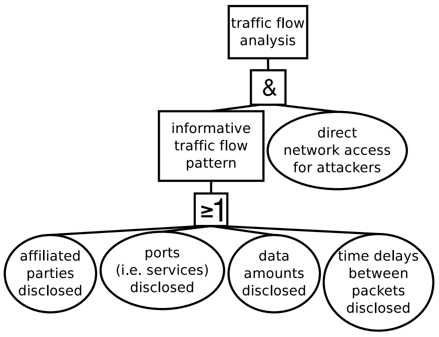

traffic flow analysis
&
informative traffic flow pattern
direct network access for attackers
≥1
affiliated parties disclosed
ports (i.e. services) disclosed
data amounts disclosed
time delays between packets disclosed

Figure 4.4: Fault tree excerpt for traffic flow analysis (cf. Fig. A.40)

cancellation of course membership can be realised online without sufficient checks, then learners could be in the position to fake cancellation for other learners to abort their participation in the course and related collaborative scenarios.

Another general approach related to this asset is given by import/export functionality, especially course exchange standards like SCORM or IMS (Instructional Management Systems) Content Packaging (CP). By using these standards, course settings and course structures can be modified outside the e-learning system, and therefore, access control mechanisms within the system could be bypassed (cf. Sec. 4.6.3).

E_3: Learning Progress and Results

Mentoring of learners requires information about cognitive barriers and emerged problems. Hence, learning progress, duration for specific learning steps, or contributions to collaborative work must be available to tutors in order to assess the situation correctly. For mentoring activities, the more information about learners and their respective problems in understanding are available, the more likely and appropriately tutors can support them. This results in demands for increased privileges to evaluate learner's activities and to get insight in personal notes. But note that mentoring can take place in different ways: either perpetually or only as demanded by learners. This corresponds to different self-concepts of learners (cf. Chap. 2, [KHS05]). While "mentoring on demand", i.e., learners by themselves ask for support in case of emerging problems, in general is accepted and desired by learners, a perpetual observation by tutors could lead to rejection (cf. Sec. 4.6.2).

Common threats to progress data can be found in general access control shortcomings to provide read or write access to persons who are not meant to access these data. This also includes tutors in case of autonomous learners who want to work on their own and only accept mentoring when asking for support on their own discretion in case of problems. Furthermore, attacks against storage media, e.g., by exploiting SQL (structured query language) injection vulnerabilities, can be used to gather progress information. Thus, data can be uncovered or even manipulated with bypassing access controls (cf. Sec. 4.7). With respect to the client-server-architecture, user interaction might also be revealed by evaluating network traffic. This means, traffic flow analysis approaches can be used to reconstruct learner activities by matching requested pages and responses to the corresponding user (cf. Fig. 4.4).

E_4: Learning Contents

Learning content is threatened by two general concepts: either it contains unreliable information, e.g., because of unauthorised manipulations, or the content is made available to unauthorised persons.

The first case of providing unreliable information can be further divided into inattentive changes by authorised persons or fraudulent changes by unauthorised ones. While unauthorised manipulations, again, can be related to improper access controls and low level attacks to base components like storage media, inattentive changes usually are related to human fallibility, e.g., entering wrong contents or including spelling mistakes. Note that especially in formulae typing errors can change the meaning significantly. Furthermore, missing support by the content management system or chaotic data structures also can provoke inattentive content changes. Missing mutual controls of authors or missing quality checks by designated quality controllers even might increase this problem.

Second, with respect to unauthorised disclosure we additionally have to distinguish the case of some external person who (actively) gains access to protected contents and the case of unauthorised distribution by a person who for that moment was authorised to access these data, i.e., copyright infringement. In case of active persons who try to gain access for themselves, threats like porous configurations and privilege settings on the server can be mentioned. This means, in case of a web server that is configured using a blacklist of file types to be prohibited for publication, further types, e.g., backup files still stored on the server, can be accessed by an attacker. Although this does not necessarily affect the most current version of protected contents, close similarities are probable, such that the attacker succeeded at least in parts. Similar to this, some web administrators seem to believe that files not linked on the web page will not be accessible to external persons. In fact, by using “fuzzing” mechanisms (cf. [MK07]), deriving hidden filenames from linked ones, or by using arbitrarily changed URLs (uniform resource locator), these files can be directly requested and accessed. With respect to unauthorised distribution, information and data that are not meant to be distributed and accessed by third parties must be protected accordingly and authorised persons should not pass these pieces of data to anyone else. This primarily concerns data with reserved copyrights as implicitly given by the sketched scenario with respect to the purchasing of created contents.

E_5: Personal Notes

Learners with respect to constructivist learning theories should be able to create their own contents during actively processing learning matter, i.e., annotations, notes, personal remarks, messages in communication facilities, submissions in exercises or self-assessments, and contributions to collaborative work (cf. Chap. 2). Especially in case of personal notes and annotations this is related to privacy and should be treated as confidential. Again, general threats by attacking the base system (cf. Sec. 4.7) and porous access control mechanisms can be the root of a set of threats related to unauthorised read or write access to foreign notes. However, besides this strict confidentiality view, also a less restrictive view on personal notes can be given with respect to sharing notes and annotations among a group of learners working collaboratively. Such special scenarios with sharing annotations and notes, therefore, have to be considered in the access control schemes. Note that threats especially will occur with respect to sharing sensitive information, if access privileges are not fine-grained enough to only provide read or even write access to a certain subset of users. An all-or-nothing approach only making contents available to anybody or no one threatens confidentiality and integrity of such data. In addition to possible technical shortcomings by too coarse privilege settings, this aspect also is related to improper default settings and missing user support.

E_6: Collaboration Data

Collaboration in the sketched scenario primarily is affected by three areas: data exchange and storage for the collaboration process, cheating attempts with respect to achieving the certificate with less efforts, and accountability for the assignment submission (cf. Fig. A.12). First, for the data exchange during collaboration, in principle two cases must be distinguished: working synchronously or asynchronously. Working concurrently on the same data, i.e., synchronous collaboration, results in problems concerning handling of conflicts. From a technical perspective, of course, only local copies of these data are used with synchronisation afterwards. But especially this synchronisation process leads to the questions of prioritisation of changes by all affiliated

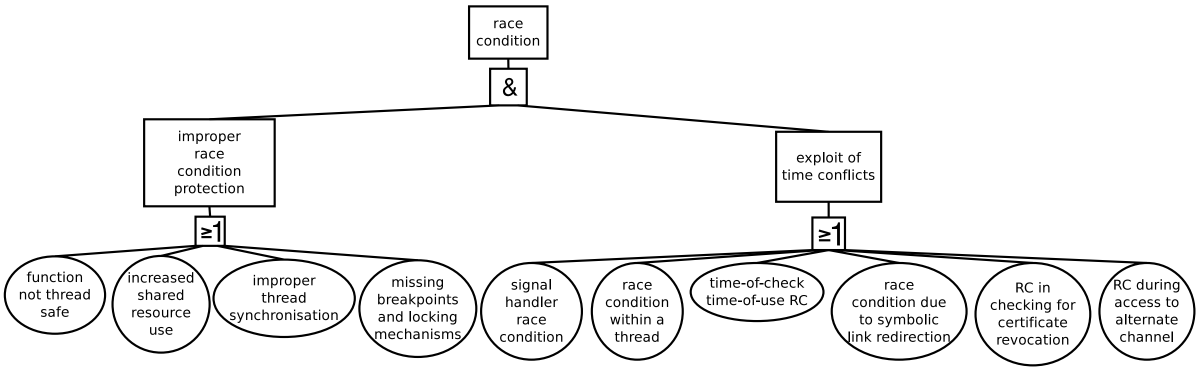

race condition
&
improper race condition protection
exploit of time conflicts
≥1
≥1
function not thread safe
increased shared resource use
improper thread synchronisation
missing breakpoints and locking mechanisms
signal handler race condition
race condition within a thread
time-of-check time-of-use RC
race condition due to symbolic link redirection
RC in checking for certificate revocation
RC during access to alternate channel

Figure 4.5: Fault tree excerpt for race conditions in concurrent scenarios (cf. Fig. A.13)

users and threats related to concurrent access, e.g., by actively exploiting competing access with so-called "race conditions" (cf. Fig. 4.5). Race conditions are comparably difficult to conduct, since timing is crucial, but they can be used to overwrite or render data of others useless by provoking temporal conflicts. In case of a missing conflict management, e.g., temporary blocking mechanisms, these threats are most likely to happen, although their existence might be difficult to prove due to a non-deterministic occurrence of this problem. The second case of asynchronous collaboration introduces threats to completely overwriting former solutions and provoking large data loss problems in case of missing history functions, e.g., by a versioning system. Since consensus among group members cannot be expected at all times, data loss by overwriting the work of others most probably will occur – either by mistake or on purpose.

Second, threats related to cheating attempts will have to be considered. In the given scenario, certificates should be issued to participants after certain collaborative results are submitted. Contents and results of other groups, therefore, might be a likely target for participants to reduce own efforts for the collaborative assignment. Weaknesses and threats related to this are given by improper access controls to current results of other groups, but also to previous (interim) results, e.g., as stored by a versioning system on the server. In addition, eavesdropping transferred data (cf. Fig. A.58) or conducting manipulations by man-in-the-middle attacks (cf. Fig. A.61) are reasonable approaches to read or alter foreign solutions. Besides this, also denial of service has to be considered as serious threat, since collaboration and submission of assignment results are not possible in case of not available services (cf. Sec. 4.7).

Finally, accountability concepts have to be assured for providing proof of having fulfilled stated conditions (cf. Fig. A.57). A major requirement of electronic submission systems is to ensure incontestable results. Thus, accountability has to be ensured in both directions, i.e., the group of learners must receive a non-deniable confirmation for successful submission – preferably with assurance that the solution was not changed during the upload process. From the viewpoint of tutors or certificate issuers a confirmation for unchanged content is essential, since otherwise learners could deny that the submitted version is the one this group has handed-in. Consequently, without appropriate accountability mechanisms issuing certificates is not possible in a sensible manner, since manipulations cannot be excluded, and therefore, legitimacy of certificates cannot be guaranteed.

E_7: Communication Data

Communication with other learners is an appropriate means to enable mutual motivation and to exchange thoughts about current cognitive barriers. This also should not exclude private communication (cf. Chap. 2). Communication channels intended for message exchange without tutors or superiors listening, therefore, need to be protected by the system. But this asset is threatened by two general aspects: restraints in the used protocol, i.e., technical limits, and direct attacks against messages.

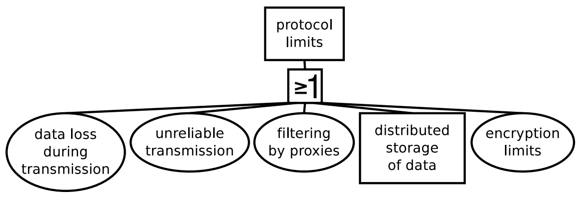


Figure 4.6: Fault tree excerpt for protocol limits in communication (cf. Fig. A.14)

A main problem with the concept of (purely) web-based e-learning systems is the aim of providing all activities and communication facilities on ground of a common technology that cannot equally satisfy requirements for all system parts. In case of web-based systems this common ground is the usage of HTTP for the entire data exchange (cf. [FGM$^+$99]). Disadvantages of HTTP (cf. Fig. 4.6), e.g., stateless operation and missing push functionality, reason a certain degree of rigidity and unreliability of the data transfer. News intended to be sent to whole groups of receivers cannot be directly transferred because of the missing push functionality and restrictions to client-server-communication. Instead, data have to be cached on the server until all intended receivers have actively requested these data from this interim node. Obviously, such caches imply the suboptimal case of spreading copies of possibly confidential messages over several systems. This threat of confidential message disclosure even gets increased in case of unlimited storage of such data, i.e., long-time data retention. Since HTTP, additionally, does not implement any sort of error detection and correction mechanisms, a secure and reliable transmission cannot be guaranteed per se. This allows several disturbances to be conducted by malicious parties, e.g., network related denial of service attacks.

4.4 Analysis of Information Security Service Assets

Information security services, as a second group of assets, are concerned with security relevant configurations and properties affecting other assets like confidentiality of sensitive data. Similar to the e-learning asset discussion, we will start with an exemplary scenario. Afterwards, threats to considered assets will be brought into relation with this scenario to emphasise practical relevance.

4.4.1 Scenario: Distant University with Various Study Courses

For the following discussion, we assume a distant university that provides various study courses to be purchased by students. The university employs teaching staff to conceptualise courses, technically experienced authors to realise these concepts within the e-learning system, mentoring personnel to support students, administrators for tasks like managing users within the system, and executives for quality controls and public relations. In this scenario a huge amount of students is considered to work simultaneously with the system in their respective courses. Collaboration and communication are essential parts of this system. Each of the courses similarly to a lecture is assigned to a single professor. In addition, roles are distinguished in global and local ones to manage groups with system-wide scope, but also to provide support for flexible role assignments and special privilege settings within a course. Thus, according to Figure 3.2, this scenario corresponds to a type 4 e-learning system.

With respect to personal expectations of not being perpetually under surveillance of tutors or professors, an appropriate level of privacy is demanded. This includes possibilities for learners to decide on their own when they want to share information about their learning progress and whether they currently want to receive support by mentoring persons within the system. However, executives particularly demand meaningful statistics for public relations to advertise provided study courses. Furthermore, evaluation phases for courses are required for quality assurance.

Evaluation surveys, of course, need to be integrated into the study process in a way that all learners may only participate once per course and stay anonymous for persons responsible for respective courses. Hence, a user's identity has to be protected for certain activities within the system with simultaneously demanding user authentication.

4.4.2 Threats of Information Security Service Assets

This subsection considers threats to assets of the information security services group (cf. Tab. 3.4). Note that these assets usually have strong connections to other assets in the system, i.e., if the assets as described in the following with related threats can be harmed, then other assets also might no longer be protected. For example, in case of stolen authentication data, an intruder can gain full access to the system, and therefore, other assets can be manipulated by abusing a foreign identity with all related privileges.

I_1: User Identity

Hiding the user identity can be sensible for various applications in the sketched scenario, e.g., for receiving more honest feedback on evaluation surveys, to enable learners to give more accurate statements without being anxious of saying something wrong, or simply to do activities without possibilities to let tutors trace unprofessional habits back to the corresponding learner (cf. Fig. A.16). Anonymity in this context can be realised in different ways, and as such, different threats can emerge to aim at disclosing the identity. Obviously, the most trivial way of ensuring user anonymity is to ignore all personal data and to not store anything that might be brought into relation with a specific physical person. But since log files and records of user activities (including personal data) are valuable for the sake of security, for statistical evaluation, and for empirical studies, anonymisation of all output can be a sensible way of hiding user identities. However, sensitive information still is contained in logs, and therefore, it could be uncovered by circumventing security means, e.g., in case of porous access control schemes, or by targeting base system components to directly access storage media (cf. Sec. 4.7). Furthermore, anonymisation cannot be considered to be completely sufficient for hiding a user's identity. Certain activities and combinations of activities can lead to patterns that can be traced back to a specific learner who should be protected by anonymisation mechanisms. Hence, loss of such data and disclosure of certain pieces of information could lead to an identification even without immediately seeing personal information. Traffic flow analysis and observing base system components together with their log files can be sufficient to extract meaningful information (cf. Fig. 4.4). Also, in case of evaluations in a small course with only a few participants, inconspicuous details like gender, age, former experience, or course of studies can be exploited to gather information about possible identities behind corresponding answering parties. Thus, anonymisation could be invalidated by specific patterns in such a situation. In addition, a special case is given by anonymous activities, where learners may only participate once. Learners obviously have to authenticate with revealing their identity while still staying anonymous. This "contradiction" can be implemented by using trusted third parties that overtake the authentication process and do not reveal the identity of an authenticated user. However, missing trusted third parties in such a scenario represent a significant threat to stated demands, since data sets, even if stored separately, could be combined as far as access control mechanisms are not implemented reliably and with proper configurations. "Trusted" in this context explicitly demands physically separated and distributed server systems with different administrators. Otherwise, administrators still could combine data sets.

I_2: Authentication Data

The authentication process is one of the most crucial processes for an e-learning system, since all subsequent security mechanisms usually depend on the reliability of this service. Thus, common threats to authentication data are bypassing of the authentication process or disclosing foreign credentials to authenticate with stolen data. Bypassing authentication mechanisms can be realised by exploiting weak protocols that allow, for example, so-called replay or reflection attacks. The former kind can be realised by eavesdropping the authentication process and sending recorded packets again to replay this process later. In the latter kind, erroneous implementations of challenge-response authentication mechanisms working in both directions will be abused, such that an attacker can send the received challenge from the server on the first opened connection back to the server ("reflect") on a second connection. The server answers this challenge with the appropriate response needed by the attacker for the first connection, such that the first connection can be established successfully. Another approach especially relevant for web-based scenarios is given by all kinds of session hijacking for bypassing the authentication process (cf. I_3 for details). Besides this, foreign credentials can be stolen or reconstructed in different ways. Credentials might be disclosed, e.g., by using so-called "phishing" approaches with providing fake log-in pages, by exploiting weak domain name system (DNS) mechanisms ("pharming") to redirect traffic, or by intercepting exchanged messages with the correct authentication server. Attackers also may be able to collect usernames if they are displayed without any restrictions within a system. Thus, only passphrases are left to be gathered as remaining part of the secret. Although the passphrase is usually better protected, it still is threatened by inadequate organisational policies, e.g., if the length of passphrases can be predicted, if patterns are used, or if no dictionary or simplicity checks are performed by the system to prevent users from choosing too weak passphrases. Even if these threats are not existent or exploitable by an attacker, brute force attacks with testing all possible combinations of usernames and passwords can be used to reveal credentials. Missing temporary blocking mechanisms, so-called "captchas", i.e., images with randomly chosen human readable information to be entered in an additional input field, or delays after wrong authentication attempts allow an attacker to apply a huge amount of combinations per second, such that successfully cracking the authentication process only is a matter of (relatively short) time.

I_3: Session Data

As introduced for authentication data, log-in status is usually managed using sessions stored on the server and accessed by clients using their session ID. Without further protection, a valid session ID stolen by an attacker is sufficient to get logged in as the corresponding user (cf. Fig. A.22). Possibilities for hijacking sessions are given by approaches like cross-site scripting (XSS), cross-site tracing (XST), cross-site request forgery (CSRF), or session fixation attacks (cf. [Tra01, MK07]). Since with Web 2.0 more and more e-learning offerings include possibilities for users to edit and comment contents, this significantly increases the amount of pages where malformed code can be injected to trick other visitors. Thus, threats related to script injections gain in importance. Besides this, in case of low entropy for the ID creation, session IDs might even be predictable for attackers. This particularly holds, if patterns can be extracted from different session IDs. But also flaws in applied pseudo random number generators used for the creation of IDs could be exploitable. Another aspect related to sessions is given by providing a store for user-related states. If an attacker is able to access these sessions, e.g., by low level attacks (cf. Sec. 4.7), then the behaviour of the e-learning system for this user could be influenced. If the system also relies on state information to be transmitted by the client, then further threats become obvious by actively manipulating corresponding information on client-side, e.g., manipulate cookie contents, hidden tags within web forms, or by injecting malware like Trojan horses. Generic manipulations during data transfer are possible by man-in-the-middle attacks like malicious web proxies, or by intercepting and altering AJAX-based requests.

I_4: Accountability Data

Accountability aims at providing non-repudiation assurance for all affiliated parties in critical activities. Consequently, appropriate mechanisms have to be present and protected (cf. Fig. 4.7). Besides generic aspects like weak authentication or session management, an error-prone privilege

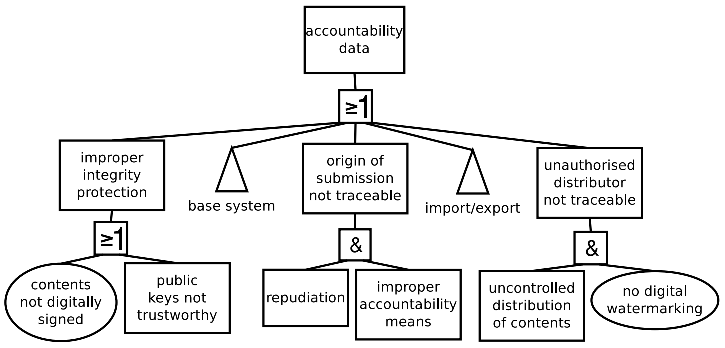

accountability data
≥1
improper integrity protection
base system
origin of submission not traceable
import/export
unauthorised distributor not traceable
≥1
&
&
contents not digitally signed
public keys not trustworthy
repudiation
improper accountability means
uncontrolled distribution of contents
no digital watermarking

Figure 4.7: Fault tree excerpt for accountability (cf. Fig. A.57)

management can be the root of improper accountability means. This means, the more users with equal privileges have access to some function, the more difficult it will be to find the actual user behind a certain activity – at least if no further non-repudiation means are implemented. With respect to mutual non-repudiation assurance and possibly mutual distrust, an additional trusted third party can be integrated, such that critical data transfer takes place in encrypted way over the trusted third party. Hence, this mediator can request and provide this information several times until corresponding partners confirm successful transmission. However, common threats to accountability, in general, are given by spoofed messages for weak protocols, network traffic redirection, or unnoticeable manipulations. Even in case of digital signatures for providing integrity and authenticity of messages, wrong public keys could be injected. Also, their validity for a specific physical person might be unsure without further identification mechanisms like provided by a PKI (public key infrastructure). In addition, low level attacks aiming at base system components or data transfer threaten accountability data, since such attacks easily can invalidate authenticity and integrity of stored or transferred data sets without being noticed at once by affected users (cf. Sec. 4.7).

I_5: Privilege Settings

As a whole, threats to privilege settings for the given scenario can be seen from two perspectives: either the privilege management is inappropriate, such that real life requirements cannot be mapped to the privilege system, or privilege settings are erroneous, such that unauthorised persons can gain direct access to desired functionality or data (cf. Fig. 4.8). Starting with improper privilege management, the set of available and manageable privileges can be too coarse to fit the actual e-learning scenario. For example, in the scenario as given in Section 4.4.1 privileges should be fine-grained enough to provide specific adaptability for integrating tutors into courses with moderator privileges. This means, although the corresponding professors or rather authors as course creators have full privileges within their courses, further personnel with limited privileges might be sensible to support learners in certain activities. If the privilege concept does not take care of such helper roles, then it might be necessary as kind of workaround to also provide auxiliary personnel with full privileges for a course. Obviously, this threatens the course integrity and confidentiality of specific settings or learning contents. Related to this conceptual threat are so-called “horizontal attacks” where users of the same role or with the same privileges influence each other and access or manipulate foreign data.

On the other hand, misconfiguration can be considered in more detail. This is mostly related to fallibility and laziness of users, such that usability and support in setting privileges appropriately are reasonable. Threats in this case are given by confusing interfaces and the lack of overview pages, such that in the example of role-based access controls no comparison of roles is supported, although this comparison would simplify the adjustment of different roles to each other. Furthermore, a lack of consistency checking mechanisms and insecure default settings increase the amount of threats related to human errors. Besides this, erroneous treatment of local role hier-

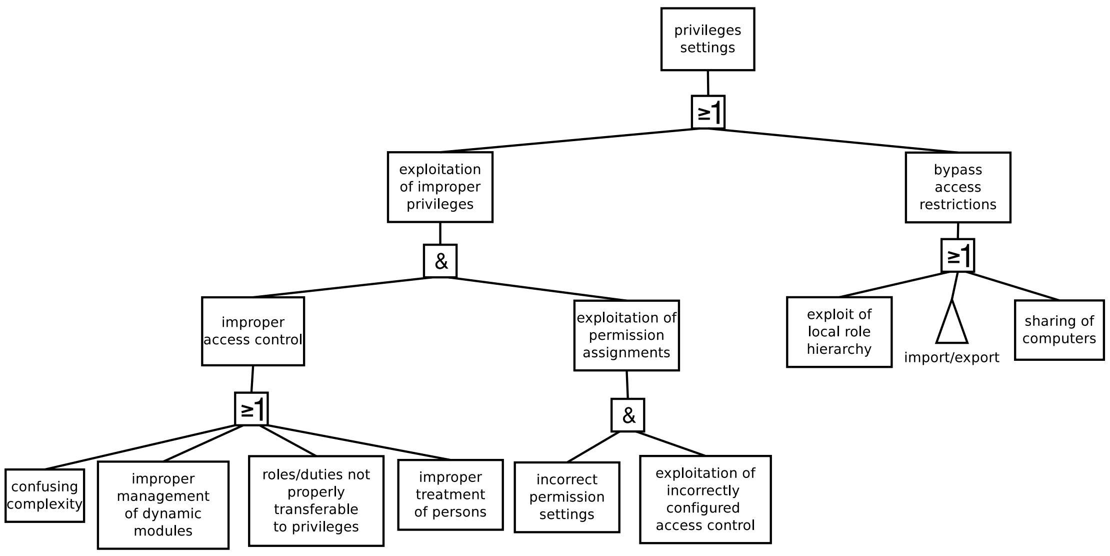

privileges settings
≥1
exploitation of improper privileges
bypass access restrictions
&
≥1
improper access control
exploitation of permission assignments
exploit of local role hierarchy
import/export
sharing of computers
≥1
&
confusing complexity
improper management of dynamic modules
roles/duties not properly transferable to privileges
improper treatment of persons
incorrect permission settings
exploitation of incorrectly configured access control

Figure 4.8: Fault tree excerpt for privilege setting threats (cf. Fig. A.24, A.25, A.26)

archies may threaten the system, i.e., access controls on a global scope possibly can be bypassed by course teachers in their local scope (cf. Sec. 4.6.1), such that local privileges can be exploited to harm other assets.

I_6: Policies and Manuals

Policies and manuals are meant to support authorised users of a system. Hence, this information should be accessible and correct; policies should be enforced to mitigate laziness or fallibility of users. Threats to these assets, consequently, are given by the lack of available information, e.g., by not created product or procedure information, by denied access as given by external denial of service attacks, or by insufficient information within provided manuals. With respect to more formal policies, clearly stated rules are desired, such that this policy can even be managed and controlled by the system itself. A problematic situation, therefore, primarily is given by a lack of formalisation, such that rules and demands cannot automatically be enforced by the e-learning system. Changes to provided manuals and policies, obviously, should only be reserved to administrators or superior personnel to prevent abuse. Especially since such documents should provide guidelines for acting correctly within the system, fraudulent manipulation, e.g., by circumventing erroneous access controls or attacking base system components, can have far-reaching consequences.

4.5 Analysis of Web Service Assets

In this section, threats to assets common for all web services will be investigated. A scenario will be introduced in Section 4.5.1 that explicitly emphasises a distributed architecture, and therefore, provides several starting points for discussing network based threats as well as base component specific attack vectors.

4.5.1 Scenario: Various Training Providers with Single Authentication

We assume an association of various software developing companies that will provide specific e-learning courses for their software. These software companies work on individual projects, but also share certain libraries to reduce their overall effort for providing high quality software. Thus, besides mutual access to work with common libraries for the convenience of customers, this association also provides a central management of account data for their e-learning courses. This includes the organisation of the authentication process as well as the administration of purchased software specific trainings. As a consequence, each customer only has to register once at this central server and at once is provided with a list of all software products for which he has purchased an educational course. These courses still are managed at the respective software company, since the decentralised content management is considered to provide better accuracy and to support keeping them up-to-date with less efforts. Courses can be purchased by arbitrary customers without any restriction. Support is not considered during the courses. Thus, this scenario describes an e-learning system of type 1 (cf. Fig. 3.2). Affiliated roles include learners, i.e., customers of this service, software companies as authors of their software specific courses, administrators for each company network to provide their own courses, and common administrative staff managing the central authentication server. Since this network in general provides distributed services, network security and reliability are of major value. The authentication process in this scenario is outsourced to a separate server for providing single sign-on for the sake of improved usability for customers. Interfaces have to be implemented appropriately to let this association of several companies work well together.

4.5.2 Threats of Web Service Assets

This subsection considers threats to assets of the information security services group (cf. Tab. 3.4). Many of the threats to be mentioned in the following will be closely related to actual attacks as discussed in more detail in Section 4.7. Hence, explanations in this place will be reduced to a minimum.

W_1: Network Topology

Distributed architectures as sketched in the scenario per se exhibit an increased amount of network traffic in comparison to single server architectures, since all affiliated services have to exchange data accordingly among each other. As a consequence, distributed architectures provide attackers with more targets and may simplify attacks. But, first, an attacker will have to uncover the actual network topology to get an impression what server systems are available and how important they may be for the architecture as a whole. Possibilities for investigating a network usually are protocol-based. Thus, the Internet protocol (IP) and related routing mechanisms can be abused to identify demilitarised zones (DMZ) or routing nodes with network address translation (NAT) (cf. Sec. 4.7). Since network communication usually depends on the domain name system (DNS) to resolve hostnames to IP addresses, an attack to this protocol, and therefore, spoofing a desired server in the network can easily be applied for preparing man-in-the-middle attacks (cf. Fig. A.61, A.67). Besides the disclosure and manipulation of the network topology with its primary servers, a recent trend to shift more and more computational processes to the clients for increased flexibility and an improved presentation by dynamic web pages, e.g., basing on Web 2.0 technologies and AJAX, results in additional targets. Although server-side systems might be protected best possible, immoderate reliance on client-side security could break the system. Considering fallibility of users, e.g., at user assisted code execution or injudicious activating of links, this also can lead to an injection of manipulated data and far-reaching consequences for user data. Increased complexity in distributing computational processes on server and clients even besides human errors can be very error-prone. Hence, critical implementation errors are very probable.

W_2: Reliability

Reliability besides the pure availability also demands efficiency, such that significant slowing-down can already be considered a problem with respect to a usable e-learning system. Thus, reliability can be seen from three perspectives: unstable software components, inappropriate maintenance including denial of service prevention, and mutual influences of deployed software.

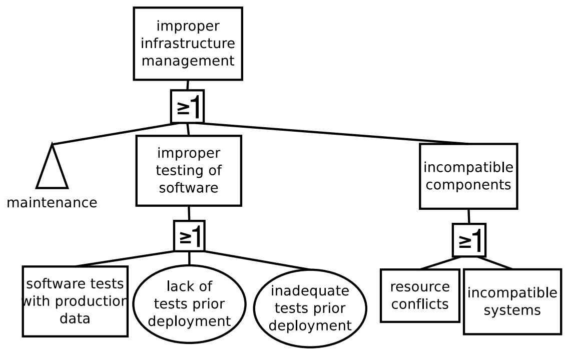

improper infrastructure management
≥1
maintenance
improper testing of software
incompatible components
≥1
≥1
software tests with production data
lack of tests prior deployment
inadequate tests prior deployment
resource conflicts
incompatible systems

Figure 4.9: Fault tree excerpt for improper infrastructure maintenance (cf. Fig. A.31)

First, due to the demand of reliability in productive environments, software has to be tested sufficiently prior to actual deployment (cf. Fig. 4.9). A flexible but due to security issues not reachable system cannot be used in any way. Hence, productive systems particularly must prefer availability and security over possible flexibility improvements by quickly applying new software releases. Nevertheless, software tests can be very time-consuming and require an experienced administrator to actively challenge the software stability and reliability of functionality. The fact of missing supervisory bodies for administrative staff – at least with comparable technical knowledge – even increases the technical problem of neglecting tests, and therefore, this can result in actually threatening the e-learning environment as a whole. Second, from a maintenance perspective, not only software patches have to be applied for closing discovered security flaws, but also adaptations to changing situations have to be considered. Delays and disadvantages by (partly) incompatible software components significantly negatively influence the performance and reliability at least of parts of the e-learning systems. Also, denial of service protection can be mentioned. Such protection consists of mechanisms being placed in the network architecture, e.g., load balancers, and respective configurations, e.g., SYN flood protection [Eri03]. Note that maintenance also includes the assurance of business continuity even in case of non-preventable incidents. Since customers paid for corresponding courses, in the sketched scenario every lack of availability could lead to indemnity claims. Threats concerning this requirement, therefore, are given by missing or permanently not available computer emergency response teams (CERT) and emergency policies in case of security incidents. Third, mutual influences usually are reasoned by exhaustive privileges assigned to processes or persons in the system, but also missing sandboxes are conducive for such threats. Similar to quarantine areas, it is advisable to run software in a restricted area not threatening any other system outside this scope. Besides the scope of software, a missing separation of storage media, e.g., by storing data of all e-learning components in the same database, introduces threats concerning the exploitation of weaknesses in one component to access data of all other components as well.

W_3: Software Details

Increasing complexity implies an increased risk of security holes to exist and to be overseen. Nevertheless, many developers tend to include more complex and general functionality. This most probably can be reasoned by the software engineering goal of reusability. If modules are implemented as general as possible, they can be applied in more scenarios than simple modules only providing very specific functionality. Thus, the inclusion of general concepts into already existing base components and the extension by additional functionality are considered to be easier than an alternative (re-)implementation of exactly those functions missing. Reusability goals and simplicity of inclusion here affect software complexity and related vulnerabilities (cf. Fig. A.43). With respect to the high probability of software containing security holes, exact information about the specific product in use and its current version is valuable for an attacker to prepare attacks with increased accuracy. Thus, threats are given by mechanisms to uncover the presence of a specific kind of software, e.g., by using port scanners, "banner grabbing" mechanisms, or by evaluating version strings as given in protocol headers when connecting to a specific service (cf. Sec. 4.7).

W_4: Interface Details

Similar to the discovery of provided software details, respective interfaces have to be protected, such that attackers must not be able to connect to these services in an unauthorised manner. Besides the pure existence of web services, specific details of provided functionality and respective protocols to access them can be considered as secret. However, transferred data can be eavesdropped if not encrypted appropriately. Open source code of services can be reviewed to extract related protocols, and closed-source products can be reverse engineered if their program files are made available to attackers. This threat especially applies for dynamic extension modules in combination with insufficient access controls. Concepts and inner implementation details of modules cannot be considered entirely in advance by administrators of the core e-learning system. Hence, an extension by modules will likely introduce security issues to be examined and exploited by attackers having knowledge about corresponding interface details for such modules. But despite the value of interface details, security by obscurity cannot be considered as reasonable security measure. Thus, gaining knowledge concerning API specifications is not considered a security breach per se, but the exploitation of interfaces as enabled by improper access controls will.

W_5: Configuration Details

This asset can be threatened in the sense of being uncovered. For example, an attacker might gather valuable information concerning weak configurations. Also, configuration files could be manipulated to inject own configurations. Since some sort of configurations are necessary for all deployed software components, several configuration files and data stores can be considered a likely target for attacks, e.g., dedicated configuration files, environment variables, or loaded and active firewall rules. Note that configuration data also could be uncovered in case of misconfigured access controls, such that remaining backup files could be requested and their contents be exploited to guess and deduce current configurations (cf. Fig. A.35). In addition, configuration details can be considered as known to an attacker if the default settings are not changed after deployment. This especially can introduce further security holes by limiting the search space for certain string values in configuration data. With respect to manipulations of configuration data, we have to consider reconfiguration, e.g., by redirecting links to libraries that get included in the e-learning core system. A possible threat for actually manipulating configuration data or at least provoke unpredictable behaviour is given by parameter injection with either changing existing values or injecting contradictory configuration entries, such that former settings might be overridden. Thus, configuration data might be exploited to clear the way for subsequent attacks.

W_6: Data Organisation

This asset has to be considered in two steps: first, the data organisation and inherent data structures will be investigated by possible intruders, and second, this information will be exploited to conduct more precise attacks. For revealing data organisation, several more or less technical threats exist. On the one hand, of course, again low level attacks and bypassing of authentication mechanisms or access controls can be applied to uncover information. On the other hand, more active attacks like the so-called "fuzzing" can be conducted to provoke errors that may contain further information about internal data organisation, e.g., database concept and names of corresponding relations or tables. Threats like enabled directory listings even enable spidering of the server file system (in respective scope) and the exploitation of search engines to find specific files. Concerning extension modules and related data storage solutions, we have to distinguish new functionality and replacement functions. Real extensions, e.g., new interaction forms and tools for collaborative learning like whiteboard integration for simultaneously working on identical representations of data, will result in new data sets that need to be protected against unauthorised access and that need to be stored reliably. Access to already existing data sets is not necessary in this case. Contrary to this, modules like the integration of a sophisticated mail system replacing a less powerful message exchange system, of course, need to access existing master data to not store such data redundantly. In addition, with direct access to such existing data sets this module can be used immediately without the need of additional time-consuming configuration. However, direct access without further configuration implies increased security issues by more probable security relevant configuration flaws. Note that this aspect concerns all assets of an e-learning system, since the actually concerned data sets depend on the corresponding module, which can provide functionality of all kind, and therefore, is not restricted to specific data sets. Actual threats related to this conceptual problem of integrating modules without strict data separation are usually of very technical kind. This mostly includes implementation errors in the module code allowing various types of code injection, e.g., by buffer overflows (cf. Fig. A.44) or SQL injection attacks (cf. Fig. A.65). Further details will be given in Section 4.7.

W_7**: Log Data and Statistics**

As discussed in Section 2.7, activities in the system are sensible to be logged and such information be combined to statistics for providing feedback for learners and for being able to reconstruct misbehaviour in critical activities. However, this requires the integrity of log files and is in contrast to user demands with respect to being under surveillance. Log data and statistics represent an asset affecting privacy and person related data. Log data, if combined appropriately, could even allow tracing a user account back to the physical person behind this account. Therefore, this asset has to be treated appropriately, such that data sets, as far as only relevant for statistics, should be stored in anonymised way. If critical activities have to be stored including personal information, then access to these data must be protected accordingly (cf. Fig. A.38). Threats related to this are given by improper access controls, excessive data logs at least increasing the impact of a successful attack, and low level attacks to base system components (cf. Sec. 4.7). Note that the primary aim of using log files to recognise fraudulent activities results in necessities to evaluate these logs and prevent misinterpretation. Thus, automated tools are needed to cope with the amounts of emerging protocol entries and support administrators in interpreting entries correctly to not harm learners by false accusations and related consequences.

4.6 Perpetual Concept Related Issues

As emphasised in the former section, certain concept related threats affect several assets in comparable manner, such that they cannot clearly be assigned to a single asset only. In the following section, three of such concepts will be investigated and discussed independently of sketched scenarios.

4.6.1 Enrolment and Role Hierarchy

Role-based access controls, despite their clear advantage for managing privileges for many users in few steps, are dangerous if applied in different scopes. In e-learning systems, mostly a distinction is realised between global scope, i.e., outside of courses, and local scopes, i.e., inside of courses. Obviously, roles and assigned privileges to roles on a global scope are superior to locally restricted privileges within a course. But with respect to these separated scopes, all roles in corresponding scopes are handled independently and describe their own role hierarchy. As long as these scopes actually are separated and each role is assigned only with privileges that do not interfere with duties and privileges of other roles, then this usually is not a problem. Note that this particularly demands an appropriate separation of duties, although in smaller institutions conceptually different roles and related tasks might be gathered in single physical persons. For example, administrative tasks may have been assigned to tutors in certain scenarios, and therefore, these tutors need to be equipped with appropriate privileges. In case of larger institutions with a vast amount of users, smaller groups with very specific duties and privileges are sensible to reduce temptations for abusing such privileges. Bear in mind that for the sake of security we always have to assume mischievous persons among users, such that privileges steadily have to be kept as restrictive as possible.

The problem of roles in different scopes becomes obvious when considering the enrolment of a user of a high global role in a course with its own role hierarchy as user of a low local role. In case of an improper separation of duties, this could introduce threats concerning unauthorised manipulation or disclosure of personal data in this local scope. For example, course creators as local administrators may access all data of this newly enrolled user – including data only relevant on a global scope, e.g., the corresponding master data. Thus, circumvention of global privilege settings could be possible by incorrectly implemented scopes. This situation even gets worse, if global administrators can be treated by such course creators or tutors just like every other user in the system, i.e., subordination by (possibly unconfirmed) course enrolment also is possible. Similar to changes of learner's profile data and personal information, changes to the data of administrators might be possible, e.g., changes to security critical parts like the degree of notifications to be sent. If only little information is available for this administrator, he cannot react in adequate time to intervene appropriately at malicious activities.

Hence, besides the basic threat inherent in several scopes for role hierarchies, missing handshake protocols for such scope changes promote mentioned threats, i.e., direct enrolment in courses must not be finished without confirmation by affiliated parties (cf. Fig. A.63). Furthermore, if manipulations and changes to the status of learners are not made obvious, traceable, and for the final steps require agreement of affected parties, then this even might occur without learners being informed in any way. Consequently, the threat of invisible and unrecognised alterations endangers users within this concept.

4.6.2 Supervision and Observation

Depending on learners' autonomy and self-concept, supervision and a complete disclosure of learning progress and former successes or failures as given by their profile might not be acceptable to learners (cf. Sec. 2.7). Hence, to know that some tutors can perpetually access these data might lead to rejection and increased distance, although the presence of moderators especially is sensible for the initiation of working processes in groups consisting of participants who did not know each other up to this moment. However, reservation and artificial behaviour could be the consequence, such that collaboration, for example, could be negatively influenced and probability of successful learning might be reduced significantly. Knowles et al. [KHS05] describe experiences with groups whose participants fall back to passivity in case of active participation and mentoring by superiors like teachers, since they completely rely on these moderators instead of taking the initiative and actively aim for achieving results.

Besides perpetual supervision, evaluation of data concerning the learning progress and possibly emerging problems also enable teachers to trace learners and disclose cognitive barriers. This means, for supervision and mentoring of learners, as much information as possible is required by tutors to support and mentor appropriately. However, conversely, this kind of data is very sensitive and might not be wanted to be shared by learners. Thus, unauthorised disclosure can be considered a security threat – even revelation to teachers, tutors, superiors, project leader, or other executives. This kind of disclosure is considered to be a threat independent of a further distribution, at least if it is not communicated in advance and related to specific supportive tasks. Note that this general threat mostly concerns the contradictory expectations of teaching

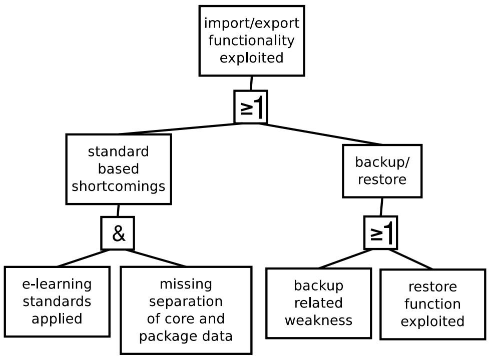

import/export functionality exploited
≥1
standard based shortcomings
backup/ restore
&
≥1
e-learning standards applied
missing separation of core and package data
backup related weakness
restore function exploited

Figure 4.10: Fault tree excerpt for import and export functionality (cf. Fig. A.48)

staff and learners. While learners usually do not want to be invisibly observed, teaching staff demands exhaustive information about learners and their learning progress do be able to support adequately. Related to this is the abuse of privileges by tutors within a course to listen to communication during collaboration. To mitigate this threat, allowing learners to hide and offer their data piecewise on their own discretion to teachers can be helpful. Privacy and acceptance of learners have to be considered to be of higher priority than supervision urges of teachers.

4.6.3 Import and Export Functionality

This general concept can be implemented in different ways, e.g., as global backup/restore mechanism, import/export of entire courses, or import/export of single components like learner profiles, list of participants, configurations, or prepared tests. Every mentioned way of using some functionality for exporting data and importing them into another system or the same system at a later point in time introduces threats concerning bypassing of access controls and the injection of malicious data. In principle, these mechanisms may provide possibilities to export data, apply changes to the archive which would not be allowed in the original system, and re-import these altered data again. Hence, by exporting information, the data are out of the scope of access controls in the e-learning system, and therefore, security mechanisms could be successfully bypassed (cf. Fig. 4.10). Affected import/export standards implemented in many e-learning systems besides a general global backup/restore functionality are given by IMS Content Packaging (CP) [IMS06a] as also integrated in SCORM [Adv09] for exchanging learning contents, IMS Learner Information Package (LIP) for exchanging learner profiles [IMS01], and IMS Question and Test Interoperability (QTI) for exchanging tests [IMS06b]. Note that especially import and export functionality including a list of participants and their learning profile can raise problems concerning privacy of learners. Consequently, the special case of complete course backup and restore functionality is not sensible for common tutors in courses with respect to privacy of learners including all information about learning progress, communication contents, and results within the learning process. For not provoking moral conflicts for teaching staff to decide between privacy of learners and better insight in learning problems, the task of managing entire course or system backups must only be assigned to administrators. Nevertheless, specific learning content backup and restore mechanisms can be integrated for teaching staff that already is equipped with privileges to legally access every single piece of exported data also within the e-learning system. But bear in mind that exporting mechanisms besides information security also affect other security areas like physical security for backup media. No unauthorised person may have physical access to corresponding backup media to prevent theft or manipulation.

Besides this, import mechanisms also could be exploited by usually authorised personnel to inject malicious code (cf. Fig. A.52). E-learning systems like all other web services have to implement proper input validation mechanisms to sanitise malformed user input and to prevent security incidents by accepting input strings without further checks. Thus, import/export mechanisms also can be abused by tutors to gain increased access to the system than usually intended by these means. For example, scripts can be injected in learning contents to realise cross-site scripting attacks or even entirely redirect users to another server under the control of the attacker. Consequently, data to be imported using such import mechanisms besides the actual authorisation of users still have to consider these data as not trustworthy. Input validation and similar protective measures need to be applied just like for every other user input.

4.7 Low Level Event Based Threats

This section investigates general threats to assets that result from (intentional or unintentional) low level events on base system components. Since events, e.g., attacks against an informatics system, are very technically oriented, many of the threats introduced in this section can be considered to be far from being specific to e-learning. They rather address web services in general. Nevertheless, e-learning systems depend on a variety of base systems like web server, database system, or operating system, and therefore, attacks to these systems affect the proper functioning of the e-learning system as well (cf. Fig. A.42). Note that many of the threats and concrete attacks to be discussed in this section were mentioned before with closer relation to the discussed assets. However, the discussion in the following will go more into detail and present actual attack procedures and possible consequences for e-learning, such that this recurring discussion aims at a more in-depth consideration.

Possible threats to mentioned assets consist of very different kinds and intentions. Numerous physical factors might threaten reliability of hardware and software components, but even more important are threats originating from human beings. Note that unintentional threats need to be considered in the same manner as the more obvious threats of an intentional kind. However, intentional attacks are much more difficult to protect against, since attackers can continually adapt to emerging security mechanisms, and therefore, perpetual analysis of the current security situation and adaptations to newly uncovered security issues are crucial.

For the following discussion, we will focus on intentional attacks while referring to the fault tree in Appendix A for respective unintentional threats. Intentional attacks can be roughly divided into three steps an intruder most probably will conduct:

1. preparation by an extensive investigation of the target system,
2. actual performance of an attack possibly including the insertion of backdoor systems to simplify later reuse and repeated penetration, and
3. “anti-forensics” by covering own tracks and make backtracking more complicated.

Especially the first step seems mostly underestimated by administrators. Skilled attackers are very patient in gaining information about a target system in order to increase the probability of penetrating successfully and of not being caught in the act:

> “A skilled intruder appreciates principles of strategy and will not rush into a system without careful preparation and planning.” [Tra01, p. 172]
>
> “When preparing to mount any attack, it is always advisable to know the terrain. In this, a skilled intruder is far from negligent.” [Tra01, p. 179]

Every piece of information that is provided by deployed systems can support intruders, and therefore, attacks to discover such information should be turned to be in vain by hiding as much information as possible or – for the sake of counterstrike – flooding the intruder with huge amounts of false information (cf. proactive defence against port scans in [Eri03]). The second step will usually take place by already prepared software tools that take care of formerly gathered knowledge about a target. In this case, in particular, it is relevant to know the operating system and hardware architecture of the target system. Adaptations to an already running attack are very difficult and mostly doomed to failure, since computers are quite fast in making decisions according to their ruleset and program logic while humans need much more time to realise their current thought and type respective commands:

> "The Hollywood-fashioned hacker that continually assaults a system login would not last an hour in the midst of contemporary firewalls and Intrusion Detection Systems (IDSs). Today's intruder is armed with an arsenal of far more sophisticated tools, which enable him to carry out more automated and intelligently planned attacks." [Tra01, p. 170]

By intelligently prepared attacks, the probability of failing in an early stage can be reduced, although an administrator of the target system, of course, will try to impede such preparation by hiding relevant information. As sort of a transition to the third step, certain software elements like rootkits can be used to keep up backdoors in the system and to cover tracks by hiding relevant information induced by activities of the intruder. Furthermore, already logged data that are related to current activities have to be deleted accordingly. Hence, from an administrator's point of view, of course, such log files have to be protected and the inclusion of malicious software be prevented.

4.7.1 Investigative Process

According to the distinction in three separate steps as introduced above, threats to e-learning scenarios can at least be divided into, first, an investigation of used software and network details and, second, the actual exploitation of the discovered situation. The third step of "anti-forensics" will be considered where appropriate in the exploitation subsection. However, as first step, an attacker most probably starts with the investigation of the target system and target network. In the following, we will give a brief overview over this general theme.

When starting an investigation of target systems and networks, besides the most obvious software interface, i.e., the provided web-based application, the base technology is of interest. This base technology promises bigger gains after successful penetration than filtered front-ends will. Hence, first, we focus on the infrastructural basis with considering threats against assets like network topology, software details, data organisation, and others (cf. Fig. A.28, A.29). A simple and useful source for information is given by the "whois" service. This service can be used to gain information about reserved IP ranges for a specific company or institution. With this information, a network scan (broadcast ping, NetBIOS name table requests, port scans, and so on) can be limited to the discovered addresses and be conducted for identifying systems online and available for external requests. In addition, a so-called DNS zone transfer, if not filtered for external requests, provides a list of all registered machines within the network. This, of course, is more precise than an anonymous set of reserved IP addresses. To further uncover connectivity and topology details of the network, protocol details like time-to-life (TTL) fields in IP headers or different appearances and compositions of received packets depending on specific implementation variants on physical systems can be exploited. With abusing TTL entries, network address translation (NAT) and load balancers can be identified, but the computers within the network are still hidden. A scientific approach that can help identifying and distinguishing such computers behind protective gateway systems is given by clock skew [KBC05, Mur06]. This approach exploits specific and individually different temperature dependent behaviour that can be analysed remotely and be used to uniquely fingerprint a system.

To gain information about the actual relevance of certain systems, and to identify services provided by physical systems, there are primarily two approaches. First, traffic flow analysis as shown in Figure 4.4 can be used to get a picture of time dependent and perpetual data exchanges among interconnected systems. This approach works best, i.e., least conspicuous, in local networks, since

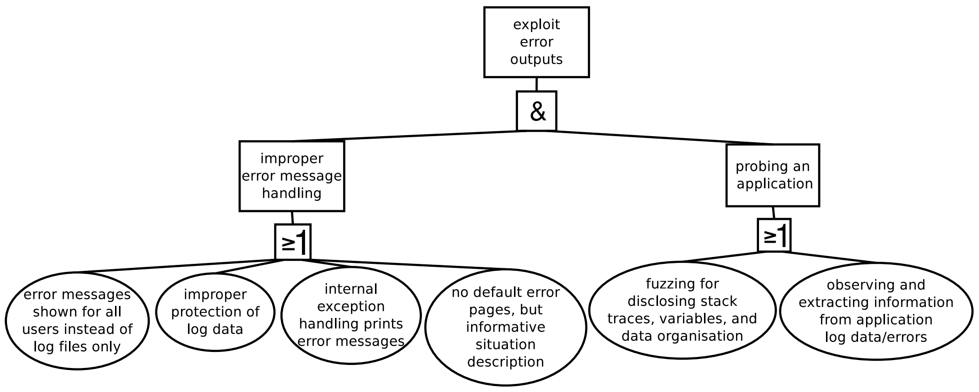

exploit error outputs
&
improper error message handling
probing an application
≥1
≥1
error messages shown for all users instead of log files only
improper protection of log data
internal exception handling prints error messages
no default error pages, but informative situation description
fuzzing for disclosing stack traces, variables, and data organisation
observing and extracting information from application log data/errors

Figure 4.11: Fault tree excerpt for provoking and abusing error messages (cf. Fig. A.34)

otherwise packets will not be visible to a passive attacker. With active approaches like ARP (address resolution protocol) spoofing/poisoning in switched networks [Eri03] or exploiting BGP (border gateway protocol) for even sniffing large parts of the Internet [PK08], of course, this restriction can be lifted. By successfully observing network traffic – even in encrypted transmissions – attackers can extract information, e.g., about affiliated parties. Computers with a lot of traffic can be assumed to be of more central relevance than machines only rarely starting requests. A second possibility to uncover relevance of systems is given by sophisticated port scanners in order to identify provided services (cf. Fig. A.33). Port scans show up which ports are open, filtered by firewalls, or closed if no service is listening on that port. In addition, port scans can be used for so-called "banner-grabbing", i.e., the scanner connects to an open port and evaluates provided information as given by the listening service, e.g., type of service and version information. Similar to banner-grabbing, significant support can be found in software advertisement strings telling almost all necessary details about the actually used software together with dynamically loaded modules and extensions, as well as specific configuration details (cf. Fig. A.35). Note that such signature strings with embedded version information are not necessarily printed on each page, but can be provoked by attackers when requesting elements that will result in error pages that usually contain this information for supporting error detection and correction (cf. Fig. 4.11). In addition to explicitly given version and software information, implicit hints can be found in specific processing behaviour, e.g., the order of header entries in HTTP responses. Furthermore, most scripting languages like PHP (PHP hypertext processor) provide information collection functions to enable developers to get an overview of what functionality and libraries are available. Due to laziness and fallibility of human developers, such information request pages often are still available in productive environments. Hence, attackers might be rewarded for guessing corresponding URLs by receiving a complete dump of configuration details and linked libraries. In case of PHP, especially option settings like "register_globals" or "magic_quotes_gpc" can be useful for attackers, since this provides information about security settings concerning parameter handling and whether "threatening" special characters in parameters are automatically "disarmed" by the interpreter.

For preparing code insertion and execution of arbitrary code, data organisation details on the server are valuable information (cf. Fig. A.37). Similar to configuration and programming errors as already described above for gaining version information of used software, error pages can also be used for collecting information about database concepts and used programming language. Concrete examples for exploiting talkative error messages related to a specific database are given by Anley [Anl02] for preparing and fine-tuning SQL injection attacks.

This list of events, of course, can be significantly extended by considering all possibilities for using software against its determined ways and misusing protocol specifications to see how the system behaves with broken requests. The introduced events, therefore, are only meant to provide a brief overview that will be picked up again in the exploitation section. Further information on technical and concept independent events can be found in [Eri03, MK07, MBPC09].

4.7.2 Exploitation

Many aspects of software and network technologies can be exploited by using them in unintended ways and by aiming at programming errors that allow malicious code insertion. Implementation mistakes provide significant potential for being exploited. Although, only a small subset of implementation mistakes can be mentioned and discussed in this section, an elaborated list of most common programming errors can be found in [MBPC09] and also in the fault tree as presented in Appendix A. As an example for a very common programming error in web-based applications, improper input validation can be mentioned (cf. Fig. A.50). This weakness, in general, refers to the processing of user input, e.g., via web forms, without having sufficiently checked input strings by the program logic including filtering or disarming of dangerous elements. As a more concrete subcategory, SQL injection attacks can be added. For this type of attack specially crafted SQL statements are entered in input fields, such that own SQL statements can be smuggled to the database and be executed afterwards (cf. Fig. A.64). This can result in unauthorised read and write access for stored data, as well as deleting entire databases or reconfiguration of database and user settings. Note that even arbitrary code might be injectable into the file system by this vulnerability and be executed on the server by exploiting extended stored procedures (ESP) [Gui09]. The relevance for e-learning systems directly results from the usual connections to a database system for storage purposes and the increasing need for enabling users to enter their own comments or similar, such that input fields are required. Related to this requirement for user input are deliberations resulting in automated corrections of user input. Fuzzy input systems are applied in certain cases to support users, e.g., to state commands in natural language, or simply for the sake of usability by not enforcing users to recall exact syntax statements (cf. [Shn92, Nie94, Dah06]). A common example for fuzzy search is given by Internet search engines providing support by also searching for similar strings not directly provided by users. However, back-end software usually needs input of parameters and commands in a very restricted syntax to be able to process this input appropriately. Assuming statements that are processed to access data in the system, e.g., input forms used for SQL statements, then a correction of wrong user input can lead to unpredictable behaviour and data disclosure, or even to data loss and fraudulent alteration as possibly desirable by attackers. Automated correction of parameters containing critical data, consequently, is not sensible and should be avoided to not threaten back-end data (cf. Fig. A.42). Note that the better the error correction is implemented, the more imprecise attackers are allowed to work with still being successful.

Famous attacks to software vulnerabilities by implementation errors are stack or heap-based overflows [One96, Eri03], format string vulnerabilities [Eri03], and exploitation of null pointer dereferences [Spr06]. All of these programming vulnerabilities are well-known to attackers and are frequently exploited. With respect to scenarios with several concurrent processes that influence each other, e.g., in collaborative work, race conditions can become problematic (cf. Fig. 4.5). This kind of vulnerability is very difficult to detect, since errors only rarely appear – at least if not initiated by attackers. Race conditions, for example, can emerge, if two or more learners simultaneously write on the same file for preparing their solutions. This can result in dropping individual solutions and only storing the last writer's data. Furthermore, temporary file management is related to race conditions, i.e., if a process needs to store interim results in a temporary file, an attacker could wait for this file to appear and change contents in this file while the main process is working on different tasks before getting back to this file again. With this approach malicious information can be inserted into the actual process.

Weaknesses related to the network layer are primarily related to insecure protocols or insecure implementations for handling protocols, i.e., protocol entries can be spoofed to influence behaviour of other technical components or even redirect traffic. For example, a spoofing approach concerning DNS in combination with cache poisoning, i.e., filling up caches with useless or malicious information, can be used to redirect requests to one's own machine and process these requests instead of the meant target. What significant impact such approaches for DNS can have, is illustrated by Dan Kaminsky [Kam08] who discovered a very critical vulnerability in the DNS protocol in 2008. Furthermore, TCP headers (transmission control protocol) and handshake mechanisms can be exploited. By guessing sequence numbers or by attentively observing transmissions with payload length and changes in sequence numbers, entire TCP connections can be hijacked and single receivers be completely replaced. Affiliated parties in such connections can be reset to not try to send any packets again to the other party. Also, TCP desynchronise can be used, such that packets are inserted with correct sequence numbers. However, due to a missing packet reception, the party to be excluded could not process and increase its own sequence number, which results in invalid packets to be sent from now on within this pending connection. In addition, software implementations that accept erroneous protocol data can perforate firewalls and be abused to smuggle malicious data into protected network areas. Protocols are very strictly specified and all states and statements that are not explicitly considered in this specification have to be treated as undefined and should be dropped or rejected. But negative examples for implementations can be found in many proxy or web servers. A common reason for system failure is given by an incomplete consideration of protocol contents, i.e., although only a limited set of commands is defined in protocols, some systems ignore or process additional statements instead of throwing errors. Mostly for the sake of performance, protocol-based evaluations are neglected. But a system in this case supports intrusion attempts by adapting wrong inputs to something that can be used within the system. For example, some header fields are only allowed once in a packet according to the specifications. But if such a field is used several times with different entries, then some implementations might also process all of them. This leads to conflicting states, since it cannot be decided by the software which entry is more reliable (first, second, ..., or last one), and therefore, this must raise errors due to specification conflicts instead of processing it in an insecure manner. Contradictory entries for the actual payload size might not be detected and treated accordingly, such that malicious data could be injected. Most famous realisations of such attacks are the so-called “teardrop attacks” using wrong offset entries in IP headers that lead to overlapping areas when re-assembling packet fragments (cf. [Eri03]). Also several HTTP exploits using an analogous technique of providing contradictory header entries for HTTP packets are commonly applied, e.g., HTTP request smuggling [Kle06] (cf. Fig. A.42). Hence, as long as error correction and fault-tolerance are implemented in this layer, this can only lead to ambiguous situations and increase the security risk significantly. For the prevention of respective threats, an abortion due to specification conflicts is sensible and necessary.

Concerning availability of services and secure connections to these services, different threats are conceivable. Besides mentioned protocol-based approaches like redirecting traffic, man-in-the-middle attacks, TCP hijacking, and DNS cache poisoning, other denial of service approaches like an increased net load, software related resource depletion, or the exploitation of programming errors to crash the corresponding software can be applied. Increasing network traffic, for example, can be realised by distributed denial of service attacks (DDoS) with a huge amount of so-called “zombie” machines simultaneously sending many requests to the target. This usually results in (legitimate) packets getting lost or not being processed due to an overload of the network or the server itself (cf. Fig. 4.12). Denial of service (DoS) attacks related to software implementation errors mostly are reasoned by other attacks that simply were not successful and instead of arbitrary code execution only led to a system crash and lack of availability.

Getting closer to e-learning systems again, concept independent events on the application layer can be considered. Here especially identity theft by stealing or intercepting authentication data should be regarded in more detail. Although several approaches exist for user authentication, e.g., challenge-response mechanisms, smart card integration, or biometrics, most of today’s authentication mechanisms still base on passwords and similar credentials. Thus, all necessary pieces of information for an authentication must be protected. In case of username/password pairs, this obviously also includes the actual username, i.e., no obvious pattern should be used for username

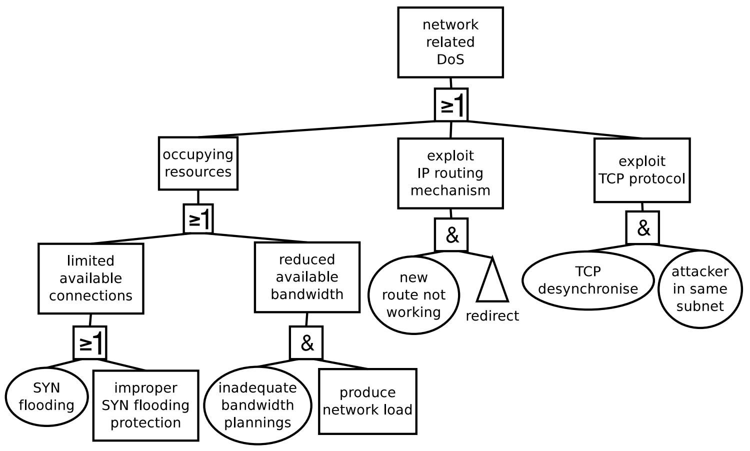


Figure 4.12: Fault tree excerpt for network related denial of service (DoS) (cf. Fig. A.56)

creation, no default usernames like "administrator", and no publicly known information as (part of) username, e.g., e-mail address. A famous attack for gaining existing usernames in a system is given by the so-called "enumeration attack". This approach exploits functions for users, e.g., account creation by themselves, password reset functionality, or the actual log-in form, to gain information about the existence of entered usernames (cf. Fig. A.20). If error messages provide information about the existence of an entered username, then the attacker has already gained one part of the whole secret and can reduce further effort to the missing elements. Since such an attack can be conducted automatically to evaluate different feedback messages, this can reveal many usernames in short time. Hence, only general error messages should be offered, i.e., "something" did not work. Note that due to mostly missing security means for ensuring a specific strength level of passwords, even simplest passwords could be chosen by users which simplifies password recovery attempts. Mechanisms for this are probing passwords by using brute force approaches or dictionary attacks. Further simplification to guessing correct passwords is given by assuming (simple) patterns in default password creation in cases where corresponding users were too lazy to change initial passwords.

4.8 Conclusion

This chapter dealt with the actual theoretical discovery and analysis of threats to e-learning systems. To conduct this analysis in a systematic way, the fault tree analysis approach was introduced and presented as the best fitting approach in comparison to other commonly used ones. However, the FTA also has several shortcomings that had to be taken care of during the analysis process, e.g., redundancy by recurring generic threats. For drawing relations to former chapters of this thesis, the analysis process was started by considering specific threats to assets as introduced in Chapter 3. Thus, specific aspects for e-learning could be discussed with discovered threats. But since e-learning systems as a collection of several activities and single services, or, as a web-based service with dependencies on different base system components cannot be covered as a whole by asset specific threats, more general and comprehensive conceptual and technical threats affecting numerous assets at once have been considered accordingly.

Threats in this analysis phase emphasised the necessity for a thoroughly elaborated security concept. Several valuable suggestions for implementing security with an appropriate level of flexibility in e-learning can easily become harmful to the system, if not considered together with related threats. As an example, role-based access controls were given with a distinction of global (system-wide) and local (course-internal) scopes. Although separated scopes with relative hierarchies provide several advantages concerning privilege management, an improper separation of duties can reverse this security concept into its opposite and even provide the circumvention of access controls by abusing these scopes.

The resulting fault tree encompassing all threats as discovered during this research project is presented in all its details in Appendix A. It consists of

- 7410 events in a fully expanded version,
- 1050 of these events are unique, and
- 597 unique events are basic events (in leafs of the tree).

These events sum up all threats as described in this chapter and provide additional threats and dependencies, which did not fit to the scenarios as used for the discussion in this chapter, but still are actually relevant for covering e-learning as a whole. Note that the difference in numbers of events in a fully expanded tree and the unique events is reasoned by the comprehensive conceptual and technical threats as mentioned in the former paragraph. Those threats can be applied to several assets in equal manner, such that they have to be integrated redundantly into the fault tree – at least by using transfer symbols to refer to those sets of threats.

Results of this chapter will be verified and extended by the subsequent Chapter 5. Given results, therefore, build a starting point for a practical examination of existing e-learning systems to complete the theoretically created tree with practical experiences and results. Afterwards, Chapter 6 will provide recommendations to mitigate or even prevent discovered threats. As outlined in Section 2.7, security mechanisms cannot be implemented by all means with respect to still enabling undisturbed learning with the system. Consequently, not all threats will be solvable entirely for e-learning, although there might exist mechanisms to protect against them and to attenuate their possible impact. Regarding technical threats, most of these events can be prevented by technical means. But what is less solvable are conceptual issues that are in touch with human users, i.e., users must be informed about security concepts and possible consequences of ignoring policies or circumventing security means. Hence, this aspect needs to be considered appropriately to find comfortable solutions in form of a compromise between users' demands and security deliberations.

Chapter 5

Case Studies

5.1 Overview

While in Chapter 4 threats were introduced and discussed in theoretical manner, this chapter aims at a practical application and verification of results. Thus, introduced aspects will be brought into relation with outlined assets for e-learning and productively used e-learning systems. In this special case, a practical analysis of threats in widely used learning management systems (LMS) will be conducted. For this, first, the term "learning management system" will be introduced in Section 5.2 together with common functional elements. Afterwards, the selection of two LMS to be considered for the further investigation in this chapter will be discussed. These LMS, i.e., Moodle and Ilias, follow rather different approaches concerning security, and therefore, enable the comparison of even contradictory conceptual deliberations. For the actual analysis in Section 5.3, a structure as given by the information security services will be followed (cf. Sec. 3.3, Tab. 3.1). Thus, mentioned LMS will be compared with respect to their different security concepts and approaches to manage corresponding security services. Conceptual shortcomings as discovered during this general analysis will be explicitly presented in Section 5.4. Also, possibilities to exploit these weaknesses will be discussed and consequences will be illustrated with their impact on respective assets. Finally, this chapter will be concluded in Section 5.5 with a discussion of results and implications for the subsequent chapter.

5.2 Learning Management System

5.2.1 Term Definition and Functionality

E-learning and learning management systems are not consistently defined, i.e., no common and widely accepted definition is available. However, the term of learning management systems (LMS) is well-spread in literature, but not used in the same meaning and not always considered to be combined with a content management component. Maier-Häfele and Häfele [MH05] introduced four different types of management systems to be distinguished: learning management system (LMS), learning content management system (LCMS), content management system (CMS), and community content collaborative management system (C3MS). This means, content management and learning management components as well as collaborative community-driven systems are considered to be different system types. Weippl seconds this view on learning management systems by his definition:

> "A Learning Management System (LMS) is software that is used for the administration of teaching and training programs. Main activities include the registration of users, tracking their progress and generating reports." [Wei05, p. 45]

Again, only the resource management facilities of an LMS are mentioned. Weippl neither considers architectural demands nor content management components, since in his opinion content management only belongs to the task sets of CMS and LCMS.

However, since in this thesis not a system-related investigation is conducted where differences of system types might be relevant, in the following, a more generous definition will be used basing on Schulmeister [Sch03b]. Learning management systems will be considered as web-based applications, i.e., digital learning environments, implementing an e-learning system in a monolithic way with functionality for administering all relevant resources for e-learning. This means, the system provides all relevant elements to support the learning process including content management. Also, organisational tasks covering all administrative, communicative, and organisational duties are considered (cf. [Sch03b, p. 10]). Due to its monolithic architecture, single elements cannot easily be exchanged, but instead they are strongly interconnected and well-adjusted to each other. An approach, alternative to combining different independent tools for building the e-learning system as a whole, is implemented in many learning management systems in form of a core system with interfaces to enable dynamic extensibility by including additional modules. Thus, this corresponds to a combined approach of a monolithic base implementing most common functionality with sufficient harmonisation and, in addition, the provision of increased flexibility by incorporating further components or by exchanging less valuable ones.

With respect to the aim of analysing practical e-learning systems for evaluating and refining theoretical results of Chapter 4, comparing different LMS is reasonable. LMS in opposition to general e-learning systems, possibly created as compounds of single functionality modules, provide a whole set of required functionality by a single instance to be installed. No further software is required; no complex dependencies to other functional software components are given. This means, loosely coupled components as different option for providing an e-learning system are quite more difficult to compare and to analyse due to missing interrelations of components and the increased flexibility for including alternative parts. In addition, numerous commercial and free LMS can be found on the Internet with connections to significant user communities. Hence, their practical value can be easily verified.

5.2.2 Selection of Learning Management Systems

In the following, LMS to be used for the further investigation will be presented. As criteria for choosing appropriate LMS as objects of comparison we use

1. availability of source code, i.e., open-source software is preferred,
2. publicity of LMS, i.e., size of community and reputation, and
3. diversity, i.e., considered LMS are not too similar.

Besides the fact of mostly being available for free, open-source learning management systems offer the advantage of allowing the source code to be analysed. This property is supported by demands of the Information Technology Security Evaluation Criteria (ITSEC) [ITS91] and the Common Criteria (CC) [ISO05] for security evaluation, since the source code as an important subject matter for the evaluation must be available for an appropriate examination of the actual strength of software. Furthermore, the openness of the source code is important for understanding extendability and implementation quality. Closed-source and commercial systems can only be regarded in a very restrictive way, since their source is not available, and therefore, cannot be included in the comparison process. The second criterion of publicity and application in numerous institutions aims at ensuring practical relevance of the comparison in the following. While the theoretical analysis in Chapter 4 can be generally applied to arbitrary e-learning systems, this chapter should emphasise security issues in existing e-learning systems with letting the reader relate mentioned threats to a system known to him/her. Obviously, this can only be achieved in case of well-spread and deployed systems with a considerable size of the user community. For the third criterion, despite the common definition of being an LMS, a reasonable degree of differences in these systems is demanded, such that different tasks and activities will most probably be solved in different manner, and as such, different conceptual deliberations can be found and discussed with respect to information security.

For the actual decision process, studies comparing several commercial and open-source LMS were applied to only consider LMS more in detail that already suffice stated requirements. These studies even showed that open-source does not give an indication for worse quality – the opposite is true. Moodle, for example, has competed with several other systems and turned out to be the best one according to stated requirements (cf. [BHM05], [MD05]). Hence, for the following comparison Moodle will be considered. As a second system, Ilias will be regarded, since this LMS also is open-source and has a significant user community. Furthermore, original ideas and concepts differ significantly from Moodle in several areas, such that the third criterion also can be fulfilled. Note that the focus on these two systems and a discussion of respective shortcomings do not mean that these systems are more insecure than others or are somewhat special in this field. The decision was made solely according to stated criteria including their publicity, and therefore, negative discussions and discovered problems are not meant to despise these systems. In fact, both systems tested very well and disclosed problems that most likely can be found in other systems as well. However, before actually comparing security concepts of these LMS, first, both systems will be introduced concerning initial aims and implemented concepts.

5.2.2.1 Moodle

Moodle[1] ("Modular Object-Oriented Dynamic Learning Environment") is a well-spread and deployed LMS that was originally initiated in 1999 by Martin Dougiamas in Australia. Dougiamas worked as system administrator at Curtin University of Technology and was not satisfied by the used commercial LMS because of problems concerning usability and support for learning. After some studies of education and early test versions, Moodle was released in version 1.0 in 2002. By now it has won a huge community of open-source programmers and users all over the world, among them the University of Siegen.

Although initially aiming at university education, Moodle is now also used in schools of different levels, non-profit organisations, and companies. Moodle enables course and user management as well as the creation of learning contents, exercises, and assessments. Besides the ease of use, the main focus of Moodle is on activity of learners, i.e., it follows a constructivist approach with emphasising social relations. It provides many possibilities for getting in contact with other learners and for interacting with the system, e.g., chats, forums, wikis, assignments, or polls to ask for learners' opinions, e.g., evaluation of course sections. Moodle does not provide the possibility of individual notes and private spaces. The often regarded metaphor of a personal desk to appear after log-in in Moodle is restricted to only showing a list of courses in which the logged in person is enrolled (independent of his role). Additional so-called blocks are available for showing small frames with information, e.g., a calendar of relevant events in courses, other persons currently online, or the most recent messages sent to a forum. Courses can consist of texts allowing HTML formatting, external pages presented in a frame, directory listings to work with files, or a collection of links to other documents. Although forums can be arbitrarily included as activity in courses, at least one forum is part of each new course for general discussions and news. Teaching staff is provided with a protected forum only accessible to them for enabling the exchange of information not meant to be visible to learners. Files can be shared and uploaded as part of a forum post to present solutions and let others discuss it.

[1]`http://moodle.org` [06-10-2009]

MOODLE is realised as a web-based application programmed in PHP and using a database back-end like MySQL or PostgreSQL. External programs for file compression in ZIP format or spell-checkers can be integrated. MOODLE organises itself in two directories, such that one directory contains all scripts necessary for running MOODLE, while the other directory holds all course data that are not directly stored in the database. For security reasons the data directory should be outside the scope of the web server to not allow upload and execution of malicious scripts. Furthermore, MOODLE is separated into a core component with scripts to manage the core functionality like administrative tasks, and modules that allow a flexible extension with additional functionality. Extension modules and plug-ins are collected by MOODLE developers and provided via the official web page[2]. MOODLE distinguishes different categories of extension modules with different ways of processing access controls. These module categories are activity modules that can be directly integrated into courses and be used as new sort of teaching method, blocks showing some information on the user interface, and filters processing entered texts. By using such filters, instead of the original input, processed representations appear in their designated places. With respect to reusability of created courses, standards like IMS QTI [IMS06b] for tests, as well as SCORM and IMS CP for the exchange of courses with other LMS are supported.

Comparing different versions of MOODLE, it becomes obvious, that not only bug fixes and minor functional extensions are implemented, but also core concepts like user and roles management as well as privilege administration are adapted significantly. During the author's research project, several major releases of MOODLE were regarded and investigated: version 1.6 (June 2006), 1.7 (November 2006), 1.8 (March 2007), and 1.9 (March 2008). As most recent version for this analysis, version 1.9.5 (October 2009) was considered. Starting with version 1.5.4 as released in May 2006, MOODLE did not implement a fine-grained access control scheme, but only fixed roles were available with their respective privileges. No further adaptations to privileges were possible. Since version 1.6 filters come with adjustable settings to adapt them more accurately to current needs and to integrate security preferences. Such settings also allow alternative back-end implementations for the expected functionality by changing configurations accordingly. In version 1.7 the privilege concept was completely re-implemented to allow fine-grained adjustments to role preferences and to overload role privileges in the scope of local courses. Although version 1.6 already introduced facilities to change the appearance of own courses to the one of learners with less privileges, in version 1.7 arbitrary roles (from an adjustable set of roles) can be chosen, i.e., different roles with different privileges and properties are available to check the course appearance from different points of view. In version 1.8 no significant changes were considered for the implementation of security aspects, but many privilege settings were tightened, e.g., while import/export of courses and backup/restore functionality was allowed for course creators in version 1.7, this now is set to the neutral "inherit" position in version 1.8, and as such, it is only available if allowed in a higher scope (not set by default). In addition, version 1.8 introduced a separation of system and site context, such that better adaptability for course internal settings is possible without affecting global preferences. In version 1.9 or more precisely version 1.9.5, again, no significant security changes were introduced, rather mainly bug fixes and new functionality. Concerning roles and their implementation, the performance was changed significantly due to a more efficient implementation of this concept. With respect to better adaptability, blocks as shown on the users' welcome screen for providing information about certain course elements and activities can be hidden from certain roles and be shown for others. Before this version, only a general publication for all roles was possible.

All these changes with respect to security and adaptability from a security point of view show that the development of MOODLE is not yet completed and the necessity for larger adaptations is still given. This aspect will be discussed again later for reasons concerning security.

[2]Modules and plug-ins for MOODLE, URL: `http://moodle.org/mod/data/view.php?id=6009` [08-09-2009]

5.2.2.2 Ilias

Ilias[3] ("Integriertes Lern-, Informations- und Arbeitskooperations-System") was initially developed in the VIRTUS[4] project at the University of Cologne, Germany, and released in 1998. The primary aim was to enable and support communication and cooperation, as well as to provide a platform for sharing information, especially to support the mentioned project. Due to positive experiences in practical use, Ilias was released from VIRTUS and developed independently as an open-source project. Currently, Ilias is used in productive environments of educational institutes all over the world – and still the University of Cologne coordinates the community of developers.

In opposition to Moodle, Ilias does not put its primary focus into activities but rather into complex course structures with many possibilities to improve course material with external links, web feeds, or a collection of easily accessible media objects. Complex course relations can be set by creating associations in form of dependencies, i.e., some courses can only be entered by learners who already have successfully passed certain preliminary courses. The session concept inside of courses provides additional structuring capabilities: files, information, or communication facilities either can be made available in a pervasive, general course element or in particular timely limited sessions. Like in Moodle, several possibilities are given for assessments and polls. Noteworthy is the integrated mail system that enables message exchanges within the system and without the necessity of making e-mail addresses available to all possible addressees. In addition, the metaphor of a personal desktop that allows the administration of courses enables an overview of current news and appointments using a calendar view. It also provides the creation and administration of personal notes to store ideas on the central server, and bookmarks to simplify access to often used elements.

Ilias is realised as a web-based application programmed in PHP. Primarily, Ilias aims at supporting MySQL as database back-end, but in more recent releases a generic database abstraction layer was introduced to also connect to other database systems like Oracle. Third party software like an image converter script, compression tools, or tools for rendering TeX formulae to images can be included. In addition, an integration of Google's map service for locating users or social bookmarking sites is possible. Similar to Moodle, Ilias for the sake of security also is organised with two separate folders, where one of them within the scope of the web server contains program scripts and configuration data, and the second folder outside the web server scope stores user data uploaded in respective activity modules. In versions 4.0.0 and newer, Ilias also provides support for an easier inclusion of additional modules, such that Ilias also can be considered as basic core component as offered by the official developers and extension modules to be installed in the directory structures of Ilias for integrating additional functionality. Another main goal of Ilias is standard conformity. Among other standards, Ilias supports SCORM [Adv09] for the import and export of learning objects, and IMS QTI [IMS06b] for assessments. Note that the conformance to SCORM even was certified by ADL[5] (Advanced Distributed Learning), i.e., conformance was verified and not only claimed as in many other LMS, although only rudimentary support can actually be found.

Comparing different versions of Ilias as released during the run-time of this research project, the security relevant improvements are less significant than those of Moodle. Starting from version 3.6.3 (May 2006) as relative object of comparison, major releases during this project are Ilias 3.7 (Aug. 2006), 3.8 (June 2007), 3.9 (Nov. 2007), 3.10 (Sep. 2008), and 4.0 (Sep. 2009). Considering the differences between these releases, since Ilias 3.7 CAS (central authentication server) and SOAP were supported for the authentication, revisions and enhancement of user registration

[3] `http://www.ilias.de` [15-01-2010]

[4] `http://www.virtus.uni-koeln.de/virtus/` [12-12-2009]

[5] certification of compliance: `http://webapps.adlnet.gov/CertifiedProducts/Certification.aspx?ID=241` [02-10-2009]

were implemented, and a PayPal[6] module was integrated for the corresponding payment method in commercially driven offerings. For version 3.8, the LDAP (lightweight directory access protocol) authentication method was extended by an automated generation of users and an import of user profiles for simplified user management. The latter task additionally is supported by automated scripts run, e.g., via the cron daemon on Unix-like systems. Version 3.9 introduced further extensions to the LDAP authentication by assigning new users to roles according to group memberships or LDAP user attributes. Ilias in version 3.10 did not implement new security-related aspects. Finally, in version 4.0 disk quotas were introduced to limit the space on the storage media for each user, LDAP and Shibboleth authentication were extended by automated assignments of local roles according to respective attributes, and a new payment method via bills was implemented together with a corresponding payment administration like storing billing address or bank data.

Comparing these changes to the amount of changes in Moodle, Ilias seems to have been much more focused on security concepts than can be found in the development of Moodle. While Moodle needed several adjustments for sufficing security demands over time, Ilias showed no significant adaptations and still provides proper security means, i.e., Ilias already had implemented required security concepts in an early version.

5.3 Information Security Services

In this thesis, the main focus when considering learning management systems will be on Moodle. This primarily is reasoned by the use of Moodle at the author's university, and therefore, by better availability of experience reports as gained due to a productive application. Ilias, on the other hand, acts as object of comparison that shows significant differences in its concept and implementation. This is mostly reasoned by a different context of creation, i.e., national legislation, planned application field, and management of further development. For the following comparison, most current releases of Moodle and Ilias as given in October 2009 are used, i.e., Moodle in version 1.9.5 and Ilias in version 4.0.0. However, former versions will also be considered where appropriate.

5.3.1 Authentication

For a proper assignment of privileges and roles to users, an authentication process is necessary. With respect to a modular architecture, local network properties, or demands for particular authentication approaches, different authentication mechanisms should be implemented in learning management systems to suit actual requirements in existing infrastructures. Nevertheless, for simplicity, in both introduced LMS a password-based authentication is used as default with storing authentication data in a database on the same server. This approach can be easily used by all learners independently of their location or the computer they are using, i.e., no further hardware requirements like chip card readers for the application of smart card authentication or fingerprint scanners for biometric authentication are stated. Furthermore, with this simple approach, no deeper knowledge about underlying technologies or complicated processes for being authenticated is required.

However, constraints given by this approach are the lack of centralised data storage and alignment, such that redundancy is necessary in case of multiple LMS, and therefore, inconsistencies will appear with considerable probability. Due to the limited length of usernames and password (for keeping it usable in practice) these data can be reconstructed in a certain period of time by an attacker. Especially in case of guessable or extractable usernames by exploring the system,

[6]`http://www.paypal.com` [17-10-2009]

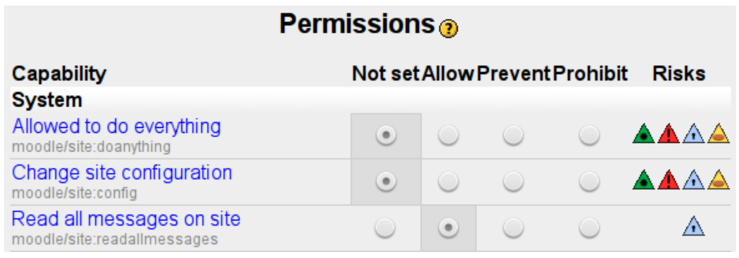
mechanisms for breaking into the system can concentrate on passwords only. Both considered LMS do not implement proper functionality for protecting against brute force password attacks. ILIAS does not block log-in attempts in any way, while MOODLE at least restricts the number of attempts to fail before a client gets blocked. This means, after ten failed log-in attempts no further tryouts are allowed, and therefore, all subsequent requests will fail despite correct credentials. But since the corresponding counter is stored in a user's session without additional consideration of client specific properties, like IP address or user agent, this only will hinder inexperienced attackers. The more common case of automated scripts to try all possible combinations of usernames and passwords can easily be adapted to renew the session cookie after a certain amount of unsuccessful requests.

Despite the simple password-based authentication approach with local storage as default setting, both considered LMS support further, more sophisticated approaches, e.g., central authentication services like Kerberos, directory services like LDAP, or Shibboleth. However, these approaches also require passwords for connecting to the central services. Alternative approaches like biometrics and challenge-response mechanisms, e.g., using public key cryptography, are not available yet in these LMS.

5.3.2 Access Control

Concerning role-based access controls (RBAC) in LMS, we are confronted with different role managements. In older MOODLE versions (before 1.7), a fixed set of roles was used that enabled a quick start using the system, but lacked further adaptability. The different roles could be assigned to users in a global scope, while still local re-assignments were possible, so that (global) students could be teachers locally. Roles were strictly hierarchically organised. In recent versions, new roles can be created and equipped with corresponding privileges, thus single roles for specific tasks are possible. Global and local roles as well as role assignments are handled differently. MOODLE has a global set of roles that can be only administered by the system administrator. Here no local roles for specific courses can be created. Instead, local assignments to roles are possible. Although the role is globally defined, certain users can be assigned to another role for the scope of single courses. In this context, functionality for overriding privileges of locally assigned roles is implemented to achieve more fine-grained access controls – if permitted by the global administrator, which is not the default setting. Hence, the general hierarchical structure of roles vanishes due to varying privilege settings. Furthermore, MOODLE enables inheritance of privileges from roles in higher contexts like the top level site or course area. ILIAS, on the other hand, gets installed with a minimal set of global roles, but automatically adds standard local roles when creating courses, e.g., course administrator or course members. Users then can easily be assigned to such new local roles. In opposition to MOODLE, ILIAS allows teachers to create new local roles on their own discretion to adjust the role setting to practical course requirements. Since ILIAS only has minimal predefined roles, like global administrator and global user, this must usually be extended prior to start working productively.

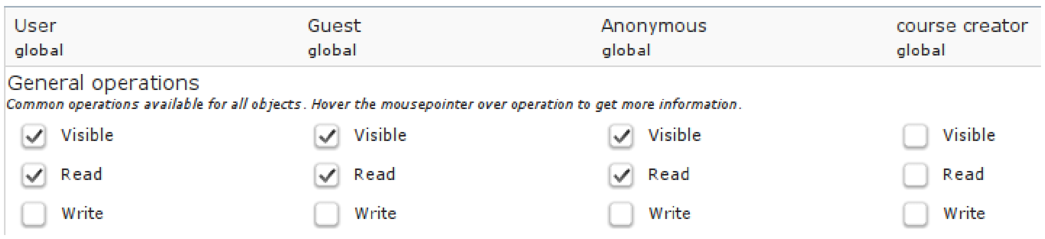
Both LMS support fine-grained privilege settings in their current versions, i.e., MOODLE has over 150 permission items to be adjusted in the main permission administration page and ILIAS about 490. Note that these numbers are strongly related to the amount of activities available in the system by default, i.e., permissions in forums are not correlated to permissions in chats or similar facilities. Hence, the set of activity facilities has direct influence on this number of permission items. Concerning the role concept itself and related privileges, ILIAS and MOODLE implement different approaches. In MOODLE roles in general are managed in a global scope by a corresponding administrator. Consequently, existing roles have relevance in all single scopes, i.e., categories and courses as well as a system-wide scope. With this concept, MOODLE implements a hierarchical system that enables overriding of privileges in higher scopes by local adaptations and, in addition, allows further settings besides explicit grant or denial of permissions (cf. Fig. 5.1). MOODLE also



Permissions

Capability	Not set	Allow	Prevent	Prohibit	Risks
System					
Allowed to do everything moodle/site:doanything	●	○	○	○	⚠ ⚠ ⚠ ⚠
Change site configuration moodle/site:config	●	○	○	○	⚠ ⚠ ⚠ ⚠
Read all messages on site moodle/site:readallmessages	○	●	○	○	⚠

Figure 5.1: Administration of permissions in Moodle, version 1.9.5



User global	Guest global	Anonymous global	course creator global
General operations *Common operations available for all objects. Hover the mousepointer over operation to get more information.*			
✓ Visible	✓ Visible	✓ Visible	☐ Visible
✓ Read	✓ Read	✓ Read	☐ Read
☐ Write	☐ Write	☐ Write	☐ Write

Figure 5.2: Administration of permissions in Ilias, version 4.0.0

supports inheritance of permissions assigned to a role in a higher context. In fact, this acts as a neutral setting in the current scope, since only higher scope settings influence the actual privilege assignment. As a fourth option, permissions can be prohibited, i.e., even if the role would be granted the corresponding permission in a lower context, this assignment is without any effect, since the higher context already prohibits this activity in all layers below. Consequently, with this option, the administrators can prohibit certain actions system-wide for specific persons without influencing local settings or other roles. Ilias, on the other hand, only supports Boolean values, i.e., either permission is granted or not. No further options are available to influence other scopes or to inherit privilege settings (cf. Fig. 5.2). But due to the strict separation of roles, this concept as presented for Moodle would not make any sense here. In Ilias roles are not only administered globally, but local roles are automatically created together with a new course. These local roles are not available on the next higher scope, such that role conflicts and interferences of permission settings are effectively prevented. Note that by the global modification of the role template of course administrators, the privilege of creating local roles can also be removed if necessary.

With respect to security usability concerns, user support in managing permissions and roles will be compared. In principle, the systems seem to focus on different types of users. While Moodle makes it very simple to find privilege settings and to get an overview which user is assigned to some role, in Ilias such settings are quite more distributed over the system. Ilias enables a more technical view on the security settings, such that identifiers for courses are applied to assign users to local roles of corresponding courses. In addition, Moodle only considers one role per time (cf. Fig. 5.1), while Ilias shows relevant roles for certain objects simultaneously (cf. Fig. 5.2). Consequently, privileges can easily be compared. An overview of existing roles and assigned privileges may increase security significantly, since it enables a thorough comparison and alignment of roles and permission settings. Therefore, it becomes obvious which roles can still execute certain activities – although the administrator, for example, wants to prevent all users from conducting them. For Ilias, this means that the administrator only has to search for the corresponding privilege and then remove all relevant checks. In Moodle, such pervasive changes would lead to the necessity of opening each role's permission table and adjusting it as necessary. This increases failure probability by obliviousness or inaccuracy. Advantageous in Moodle, on the other hand, is the fact that it marks permission items and gives hints about possible consequences of wrong permission settings. In the right-most column in Figure 5.1 there are four different icons that indicate impending threats like "users could change site configuration and behaviour" (green), "users could add files and texts that allow cross-site scripting" (red), "users could gain access to private information of other users" (blue), and "users could send spam to site users or others" (yellow). Note that not all permission items are yet marked with such threat indications, i.e., warnings are not completely available and users, therefore, are partially left on their own.

5.3.3 Data Confidentiality

Besides the common recommendation of both systems to apply encryption for transmitting data, i.e., using HTTPS, they handle data related to confidentiality quite differently. Considering default privilege settings of both systems, Ilias seems to be focusing more on privacy and individual decisions. This means that teaching staff is not in the position to administer learner accounts and to access all of their data. In general, users can adjust on their own, which data of their profile should be visible to others. Thus, although all necessary data are given in the system, they cannot be extracted by unauthorised persons. Moodle, on the other hand, has some flaws in this context, since teachers are considered superior to learners, such that – even in case of global denial of access to some learner's profile for a teacher – such a teacher as course administrator can easily enrol the learner in one of his courses and afterwards will have read and/or write access to the learner's profile (cf. Sec. 4.6.1 and 5.4.1).

In addition to explicitly collected data, e.g., user data stored in a profile, we have to consider implicit data logs. Schulmeister considers such access logs and activity recordings as extraordinarily important, since in his opinion teachers, by using such mechanisms, can gain performance information of course participants, and therefore recognise whether problems occurred [Sch03b, p. 8]. Activity logs, in the sense of empirical data, in certain degrees support in evaluating courses concerning their popularity and utilisation. Furthermore, activity recordings provide information about access to certain files, such that teachers can be notified whether all learners have already accessed corresponding contents. Both, Moodle and Ilias, provide logging capabilities. While Moodle considers statistics as optional and allows them to be turned off, activity logs are always written – only visual evaluations in the form of a statistical representation are not available in that case. Anonymisation is not considered in Moodle. Ilias, on the other hand, provides separate settings for activating user tracking by log files and evaluation by statistic capabilities. In case of activated tracking functions, Ilias furthermore provides an option for only storing data in anonymised way to reduce privacy concerns. Note that all such tracking functions are turned off in Ilias by default. Besides general logging mechanisms, both systems provide functions for keeping track of last user access to the system. This function is needed for maintaining a list of currently active users in the system, i.e., a list of who has conducted an activity within a certain period of time before the corresponding request. Thus, this list presents potential partners for synchronous communication, since they are possibly still logged in and available using the e-learning system. If data about the last access of users to certain contents are available to teachers, then this can also be abused to gain information about the learner's utilisation of the system. With regard to the mentioned LMS, despite the fact that both systems maintain such access logs, their publication of respective data is different. Moodle shows the date and time of the last access to the system in activity logs as well as in corresponding user profiles. Ilias provides this information to administrators only.

With reference to E_7, Section 4.3.3, Moodle and Ilias as web-based monolithic systems suffer from the inappropriateness of HTTP as a basic protocol to be used for data transfer of all kinds, including communication data. Communication data in this case have to be cached on the e-learning server, e.g., by storing them in the central database. These data then can be requested several times from authorised users until they get deleted. Since databases are likely targets of attacks, e.g., by SQL injection, storing data over long time could possibly lead to the disclosure of private contents, and therefore, to confidentiality breaches. To restrict data retention, Moodle and Ilias use scripts that can be run frequently by the operating system to delete messages that are older than a specified number of days. This number of days in Moodle, for example, can be chosen from a drop-down menu when creating a new communication activity, i.e., no arbitrary, but only predefined durations are possible. The default setting is to never delete these messages. Consequently, at least the administrator (and successful attackers) can read all transmitted messages directly from the database (cf. database table "mdl_chat_messages" for Moodle). Besides communication data, this problem exists for almost all modules and activities of HTTP-based learning management systems.

5.3.4 Data Integrity

Eibl, von Solms, and Schubert in [ESS06] presented an evaluation catalogue to roughly audit the security of existing e-learning systems by simply checking the presence of protective measures for each information security service. Having applied this catalogue to Moodle and Ilias it turned out that in both systems, besides counting on an encrypted data transmission and sufficiently secure authentication mechanisms, no further integrity protection like an application of digital signatures is implemented. Consequences were discussed by the same authors in [ESS07] and respective results will be presented in Section 6.6. However, a conceptual deliberation actually threatening integrity in Moodle is given by the possibility to log-in as different (subordinate) user. Moodle allows persons with higher global roles to log-in under the account of a participant of one's own course. This option was probably meant to support learners with technical problems, so a teacher can get the same view on the system as the learner, and therefore, can give more accurate advice. Furthermore, this provides the ability for teachers to see their courses in the same way as learners. This has the advantage of getting better insight, as to whether a course is designed appropriately or whether certain elements are confusing, or even missing. Schulmeister demands this "debug functionality" even with a value of 5, which is the highest possible rating in his catalogue for necessary criteria [Sch03b, p. 59, tab. 15]. The problem related to a complete log-in change is a much simplified identity theft, which can be considered to be very critical. Teachers can read all private notes, profile entries, or act under the name of that learner and write messages to others. Although the Moodle developers did mitigate this problem, and in current versions only provide this option for administrators and not teachers, this option is still part of the system. A better solution for changing views was introduced in version 1.6 with changing roles in the system, i.e., in Moodle 1.6.x rather than the teacher's view, a student's view can be chosen. And, in versions larger than 1.7, all subordinate roles can be chosen for changing views. This means that teachers can drop privileges and view their courses with the privileges as given to learners. Ilias has also implemented a change of view for its courses with the approach of dropping privileges.

5.3.5 Non-Repudiation

Similar to data integrity, none of the considered LMS implements appropriate mechanisms for proving authenticity and guaranteeing non-repudiation of origin or receiver. Only logging of all activities and data transfers including request and response details like source and destination are implemented in Moodle and Ilias. Note that for web-based e-learning systems, the dependencies on other systems like web servers, database systems, or the operating system result in connected logs of these base components, i.e., each of the components will record activities targeting the respective system. Thus, a combination of log files can give extensive information about activities within an e-learning system. With this, activities can be reconstructed and originators can be identified – at least if the integrity of these data can be assumed. But note that data entries, as given by such logging mechanisms, only contain information being received by the base system components, i.e., manipulations on their way to the server or client like those given by man-in-the-middle attacks, will not be recognised by server-side non-repudiation mechanisms only. Hence, non-repudiation is not incontestable in mentioned systems due to the lack of client-side mechanisms such as applying digital signatures.

5.3.6 Availability

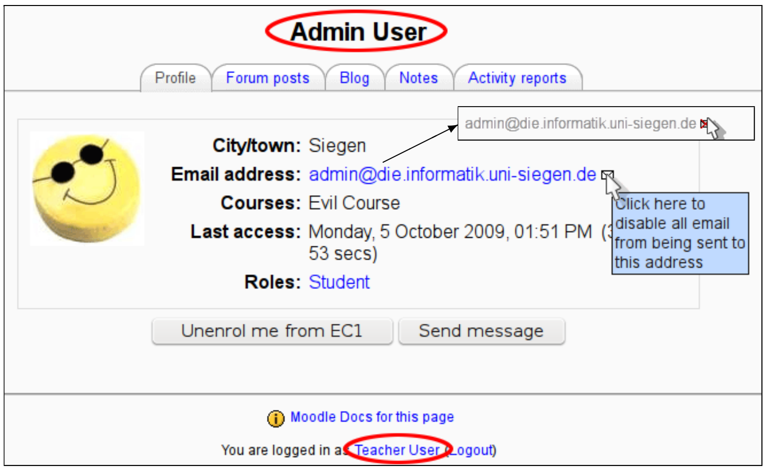
The service of availability is primarily related to the technical basis and base system components. Nevertheless, certain functionality in the e-learning system is supportive, e.g., backup/restore mechanisms for being able to restore relevant data quickly in case of security incidents or hardware failures. Moodle and also Ilias provide appropriate functionality to take backups of courses. This means that teachers and course creators can backup learning contents of their own courses, while administrators are provided with the functionality for taking backups of the entire system. This would include user accounts with related data, configuration settings, and uploaded files. Further functionality implemented in both regarded systems concerning availability is given by multiple confirmation dialogues in the case of critical activities, such that users are getting warned to avoid unintentional changes of critical settings and the conduct of critical activities.

5.4 Conceptual Shortcomings

The investigation of Moodle and Ilias revealed several conceptual weaknesses. These weaknesses already were sketched in Chapter 4 in a theoretical manner and will be discussed concretely in the following. Note that presented weaknesses are by design, such that it is much more difficult to solve these problems compared to simple implementation errors where patches could be quickly applied. Instead, whole implementation concepts need to be changed and a thorough code review might be necessary.

5.4.1 Deactivation of Foreign Mail Addresses

Roles in learning management systems usually have an inherent hierarchical order. For example, if assuming distant universities, then administrators have more privileges than teaching staff and all members of teaching staff have more privileges than learners or even guests. Roles cannot only be considered in connection with physical persons, but also with respect to dependencies on course settings. This implies that teachers could enrol other users in the system into their own courses and subordinate them locally as learners (cf. Sec. 4.6.1). As far as the LMS does not consider this scenario and even rewards such activities by teachers with actually allowing them to access confidential data of enrolled users, confidentiality and integrity with respect to unauthorised administration of personal data are affected. Ilias is prepared for such a scenario and only allows course specific changes to data, i.e., different scopes are logically separated. Moodle in older versions (before 1.7) allowed the disclosure and alteration of profile entries of (local) learners. In recent versions this is disabled by default, but Moodle still allows the deactivation of e-mail addresses of "subordinate persons" (cf. Fig. 5.3). This probably was meant to prevent bounce mails in case of obviously no longer existing mail addresses. However, this feature also can be abused to exclude affected persons from receiving informative messages of all kind. Note that this change has system-wide effect and is not bound to a specific course. So, if the administrator can also be enrolled in one's own course just like every other user in the system, then this has significant impact on the security of the system as a whole. Administrators need to be available and be informed by the system in case of incidents or system failures of all kind to react as soon as possible and to be supported by as much information as necessary to act appropriately. The basic problem behind this conceptual weakness is the mixture of local scope role hierarchies and global scope settings. Furthermore, a missing handshake for course enrolment



Admin User

Profile | Forum posts | Blog | Notes | Activity reports

admin@die.informatik.uni-siegen.de

City/town: Siegen
Email address: admin@die.informatik.uni-siegen.de
Courses: Evil Course
Last access: Monday, 5 October 2009, 01:51 PM (53 secs)
Roles: Student

Click here to disable all email from being sent to this address

Unenrol me from EC1 | Send message

Moodle Docs for this page

You are logged in as Teacher User (Logout)

Figure 5.3: Deactivation of administrator's mail address by teacher

and transparency at least for enrolled users about their changed status and assignment to another course even makes this problem worse.

5.4.2 Personal (Contact) Data in Search for Users

For contacting users because of administrative reasons, contact data must be available and be stored in the system. But these data are not relevant for all parties in the same manner, and therefore, access has to be limited to authorised persons only (cf. E_1, Sec. 4.3.3). Concerning usability and user interface design, according to Shneiderman [Shn92] users always should get the feeling of having influence on the system and being able to control the data flow. ILIAS, for example, provides fine-grained sharing possibilities for personal data in user profiles, such that a user can explicitly hide or show certain information. Further data sets can still be stored in the profile without being visible to the majority of other users. However, a conceptual shortcoming in MOODLE also results in a leak providing users with a vast set of user-related data – in this case name and e-mail address. Bear in mind that confidentiality leaks providing e-mail addresses to third parties primarily can have negative consequences with respect to unsolicited bulk mail. With holes in the system, a huge amount of e-mail addresses of registered users can easily be gathered. Connected to this, another threat is given by the application of these addresses for sending faked messages from the LMS to conduct "phishing" attacks to collect further user specific data. In MOODLE, for example, the search for users can be considered vulnerable with respect to such data loss. Users for course enrolment are identified by a combination of their real names and their e-mail addresses. Therefore, entered search strings are applied to this concatenated string. Thus, all combinations matching the search string are presented as results – including the entire e-mail address without being scrambled. This mechanism, of course, can be abused for collecting a huge set of addresses, e.g., by searching for common e-mail providers or character combinations including the @-character. This problem gets even worse when considering the fact of learners also using their private e-mail addresses instead of being forced to use, for example, addresses provided by the actual educational institution or company maintaining this LMS installation. Addresses of the latter kind usually are temporarily limited and could be created especially for being used in this online scenario. Consequently, other addresses would not be affected in case of unauthorised disclosure and abuse. In addition, by using corporate addresses the corresponding institution is in the position to support their users by sophisticated protection mechanisms – such as the most up-to-date filter technologies or malware protection.

5.4.3 Generalisation and Module Inclusion

In this subsection the so-called "TeX filter" of Moodle will be considered as an example for vulnerabilities related to module inclusion and design decisions towards more general functionality. This input filter is meant for rendering text describing a formula in TeX notation (surrounded by double dollar signs). As an example, the binomial theorem entered in Moodle as

```
$$ (a + b)^n = \sum_{k=0}^{n} {n \choose k} a^{n-k} \cdot b^{k} $$
```

will be rendered automatically by this filter to an image showing the semantically equal formula:

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} \cdot b^k$$

Images created by this filter get integrated into the HTML pages of a course instead of their original textual representation as entered by the author of the learning content. This enables the usage of mathematical formulae in courses, which is especially necessary in topics of natural sciences or engineering. However, note that the filter is available to all users of the system after being activated by the system administrator. No further privilege settings for fine-grained access control are possible in Moodle.

Interestingly, within only a few months in the mentioned filter three critical security holes were discovered and published:

1. remote code execution in file "texed.php" (Oct. 2008),
2. sensitive file disclosure using LaTeX back-end (Mar. 2009), and
3. buffer overflow in mimeTeX back-end (May 2009).

The first vulnerability was found in October 2008 and published in December 2008 using the BugTraq mailing list (provided by Security Focus [POP08]). This security issue concerned a parameter for describing a path in the file "texed.php". This file has no essential meaning for the filter and its proper processing of entered text. It is meant only for authors to test their syntax of a formula to check it prior to the actual integration into learning contents. The security issue requires the option "register_globals" to be turned on for the PHP processing module, although this option is already off by default since PHP version 4.2 (April 2002) for the sake of security.

The other two vulnerabilities target the back-end system for actually rendering LaTeX formulae. Moodle is shipped with a version of the "mimeTeX"[7] application, which implements a very limited set of LaTeX commands for processing formula notations. Thus, vulnerability three is given by implementation errors in this application (cf. [Bar09]), such that long input strings could lead to a buffer overflow possibly enabling the execution of arbitrary commands on the server. In the following, the second vulnerability will be considered in more detail. This security hole was discovered by the author of this thesis in March 2009 and was promptly reported to the Moodle developers (cf. Moodle bug MDL-18552, MSA-09-0009[8], and the BugTraq report [Eib09b]). This vulnerability is reasoned by an improper input validation, and in addition, by a decision towards increased functionality. Since Moodle version 1.6 (June 2006) a complete LaTeX environment on the server is preferred over the supplied mimeTeX. Although the command set of mimeTeX for formula descriptions should be sufficient in most cases, the more powerful LaTeX environment is preferred, probably due to possibilities for including further symbols, redefining terms for providing shortcuts, and even processing complex structures for the resulting output. But this generalisation leads to a significant increase in complexity. LaTeX as back-end system for the use in the input filter represents a Turing complete language that even supports file system

[7]`http://www.forkosh.com/mimetex.html` [08-06-2009]
[8]`http://moodle.org/mod/forum/discuss.php?d=121039` [04-10-2009]

operations in certain borders. As consequence, besides possibilities to read confidential data, an attacker might even be able to inject arbitrary code under certain circumstances. However, the latter case of code injection will not be further investigated due to the restrictive default configuration of TEX for file output (cf. option "openout_any").

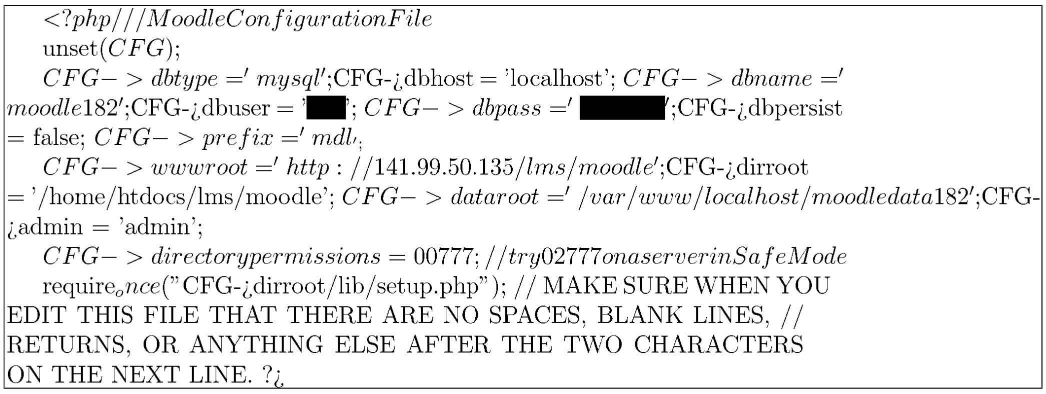
The generalisation and the urge for more flexible and universally applicable modules in this case led to an ignorance of security considerations. On the web page of mimeTeX, explicit warnings concerning security risks related to certain LATEX commands can be found. This also indicates an implementation of mimeTeX with certain restrictions:

> "\input{filename} behaves just like the corresponding LaTeX command [...] Moreover, for security, absolute paths with leading /'s or \'s, and paths with ../'s or ..\'s, are not permitted." (`http://www.forkosh.com/mimetexmanual.html`; 19-03-2009)

LATEX as indicated in this quotation supports the function "\input" for including external files. However, the security mechanisms as mentioned for mimeTeX by default are not active in the LATEX environment due to a typically different application field in comparison to mimeTeX. An attacker, therefore, can access confidential data with sufficient knowledge about path structure and LATEX syntax. This means, an input string like

```
$$ \input{../../../htdocs/lms/moodle/config.php} $$
```

can result in an image with sensitive information like illustrated in Figure 5.4.



<?php///MoodleConfigurationFile
unset(CFG);
CFG− > dbtype =′ mysql′;CFG-¿dbhost = 'localhost'; CFG− > dbname =′ moodle182′;CFG-¿dbuser = '[redacted]'; CFG− > dbpass =′ [redacted]′;CFG-¿dbpersist = false; CFG− > prefix =′ mdl′;
CFG− > wwwroot =′ http : //141.99.50.135/lms/moodle′;CFG-¿dirroot = '/home/htdocs/lms/moodle'; CFG− > dataroot =′ /var/www/localhost/moodledata182′;CFG-¿admin = 'admin';
CFG− > directorypermissions = 00777; //try02777onaserverinSafeMode
require$_o$nce("CFG-¿dirroot/lib/setup.php"); // MAKE SURE WHEN YOU EDIT THIS FILE THAT THERE ARE NO SPACES, BLANK LINES, // RETURNS, OR ANYTHING ELSE AFTER THE TWO CHARACTERS ON THE NEXT LINE. ?¿

Figure 5.4: Example: disclosure of MOODLE configuration file

This configuration file, as uncovered by abusing the TEX filter, clearly shows on which server the database used by MOODLE can be found together with all relevant authentication credentials (blacked in the figure) and the database name. Thus, if database access over the network is supported by the corresponding network architecture, then the attacker can directly access and manipulate the entire database including the creation of a new user account with administrative privileges or the disclosure of confidential communication contents. Note that besides the local configuration file of the LMS, all other files on the server can also be read as far as they are accessible with the privileges of the web server. This includes operating system files like "/etc/passwd" for investigating existing user accounts on the server and all files uploaded by using MOODLE routines, e.g., files attached to forum posts.

5.5 Conclusion

In this chapter, we compared the open-source learning management systems MOODLE and ILIAS. Besides an initial discussion of the term "learning management system" and introductions to

Moodle and Ilias, including historical development of security concepts and general functionality, the comparison was structured according to the six information security services as introduced in Chapter 3. Although both systems showed similarities due to their common property of being LMS, they showed also significant differences concerning their aims and security concepts, which provide disadvantages and advantages with respect to accordingly considered situations.

Moodle and Ilias show significant deficiencies concerning protection against brute force attacks for authentication data. Neither delays nor temporary blocks of user accounts are implemented. Furthermore, the security services of data integrity and non-repudiation are not considered sufficiently in their security concepts, since these services almost entirely rely on the authentication process and on logging mechanisms to reconstruct user activities. Digital signatures or similar encryption-based mechanisms are not implemented. In addition to these security services, more general conceptual shortcomings were presented as primarily found in Moodle. Among these shortcomings, the circumvention of access restrictions by poorly conceived role and privilege management facilities was discussed. The basic principle of generalisation of functionality in combination with dynamic extensions by modules was considered in detail with an example introducing several vulnerabilities. Since conceptual shortcomings are by design, this obviously emphasises a significant need for a thoroughly deliberated security concept.

Nevertheless, both regarded LMS, despite the mostly insufficiently secure default settings, provide more sophisticated and adequate security settings, which can be adjusted accordingly for productive environments. In general, Ilias seems to focus more on adult learners with an advanced demand for self-direction in learning. On the one hand, this is supported by anonymous communication channels and possibilities to prevent exhaustive logs. On the other hand, several user controlled settings for sharing individual data like personal notes and for providing specific contact information like instant messenger ID are provided. Such publication is left to the individual user's discretion and is not set in a mandatory way by superior administrative staff. Concerning usability and support in administrative tasks, Moodle offers helpful information in the form of coloured symbols and textual hints for items in the privilege management page. This can be helpful to mitigate inattentive changes by less experienced administrators, since possible consequences of misconfiguration get illustrated and described prior to actually confirming such threatening settings.

In the following chapter, recommendations will be presented to cope with mentioned theoretical threats as introduced in Chapter 4, but also for conceptual shortcomings and technical weaknesses as presented in this chapter for regarded LMS.

Chapter 6

Recommendations

6.1 Overview

For managing risks, first of all, it is necessary to know about their existence and possible impact [Sei06]. This includes the identification of threats as discussed in Chapters 4 and 5, and implies deliberations for conducting appropriate management steps. Hence, this chapter deals with recommendations to mitigate discovered threats (cf. Fig. 6.1).

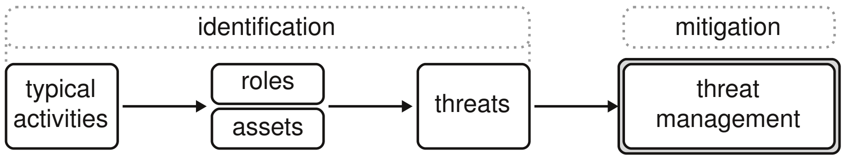


Figure 6.1: Current position in analysis process – step 4

For this, risk management strategies will first be discussed in order to distinguish different ways of treating threats (cf. Sec. 6.2). Note that in general not all threats will be solvable in equal manner – some will even have to be accepted to a certain extent due to missing reliable solutions, e.g., human fallibility.

> "Often, the term security solution suggests the security risk has been eliminated. Nothing could be further from the truth. Not one single security solution has ever proven to be perfectly secure. And, for a variety of reasons, none ever will. All solutions are relative." [MC01, p. 32]

Nevertheless, even if no satisfactory solution is available for some threats, at least the consciousness of still existing threats can be regarded valuable, since administrators can be prepared to detect attacks and initiate counter-measures. However, for actually solvable threats recommendations will be presented with numerous references to the fault tree as depicted in Appendix A to draw relations of recommendations to respective threats. The remainder of this chapter is organised following the "layer" dimension of Figure 1.1 on page 4, i.e., (1) technical basis representing essential system components like the infrastructure and functional elements (cf. Sec. 6.3), (2) formal policies including conceptual aspects and regimentation (cf. Sec. 6.4), and (3) the users of the system (cf. Sec. 6.5). Each of these sections is constructed like a set of simple rules and recommendations with brief explanations. In addition to those sets of recommendations, in Section 6.6 a proxy server realisation as security agent for overtaking security tasks and for relieving users from doing them will be presented in detail. This agent, of course, will take care of design criteria (cf. Sec. 2.5) and limits for security mechanisms (cf. Sec. 2.7), such that it can easily be integrated into existing e-learning infrastructures. Finally, this chapter will be concluded by summarising and discussing presented concepts and implementation aspects.

6.2 Management Strategies

Different management strategies are conceivable for mitigating risks (cf. [SGF02, Sei06]): avoid or at least limit risks by implementing appropriate security means, "transfer the risk by using other options to compensate for the loss, such as purchasing insurance" [SGF02, p. 27], or simply accept a risk that cannot be handled otherwise in reasonable manner. Note that although in this thesis probabilities of threat occurrences for estimating risks will not be examined (cf. Sec. 1.4.3), threats as base conditions for the existence of actual risks can be considered with comparable management strategies. Realistically, not all threats can be avoided and not all problems be eliminated, although this would be desirable for the sake of security. Especially comparisons of costs in case of security incidents, and costs for mitigating risks reasoned by concrete threats should be considered to accurately decide between different alternative strategies. In many cases, security mechanisms are introduced in practice without even considering their negative impact on the actual task of the system. But, as stated by West [Wes08], security in best case is of secondary interest. The primary task, i.e., the task a user actually plans to do with the system, should have priority. However, security mechanisms should assist and protect this task. In case of very restrictive security mechanisms, obviously, the primary task cannot be conducted in all situations with the needed flexibility and functionality. Hertz and Thomas claim that with respect to economic systems, too strong security mechanisms and systems for eliminating risks "result in the greatest risk of all: rigidity" [HT83, p. xii]. Anderson seconds this claim and encourages to think of more appropriate alternatives to cope with identified threats:

> "The point is, don't fall into the trap of believing that the only possible response to a vulnerability is to fix it; and distrust the sort of consultant who can talk only about 'tightening security.' Often, it's too tight already." [And01, p. 490]

However, with respect to the sketched e-learning types (cf. Fig. 3.2, p. 33), no general solution and set of recommendations can be given fitting to all e-learning systems in the same manner. Introduced types of e-learning systems, obviously, are different with respect to the complexity of the system, interconnection of physically separated components, emphasis on security or usability and user acceptance, and possible side-effects influencing other system components or users, e.g., concurrent work on the same data sets. Hence, it is sensible for different management strategies to be applied for the actual type and related characteristics and priorities concerning system functionality and emphasis. Bear in mind that the threat analysis as presented in this thesis considered characteristics of all e-learning types, such that the resulting fault tree covers particular requirements and conceivable threats of all these types. Analogously, in the following, recommendations will be presented that are based on the fault tree, and as such, also are in touch with characteristics of different types of e-learning systems. This means, threats as revealed in this thesis were examined with conceivable counter-measures, such that reasonable means in form of recommendations could be deduced for further discussion in this chapter. For the actual practical application of findings within the analysis or introduced recommendations, therefore, conceptual deliberations, as well as adjustments according to the effective lists of services of the considered e-learning system, must be taken into account. Since not all services and investigated functionality components are available (and needed) for all types of e-learning systems, only a subset of these recommendations might be directly applicable to a reader's infrastructure or e-learning system. Others will have to be (considerably) adapted, and some of them will not be applicable at all due to services not being implemented. Obviously, recommendations to absent services can be ignored. However, in this chapter we assume all introduced services to be of importance, such that recommendations can be introduced and discussed accordingly.

Table 6.1: Technical aspects and requirements for secure e-learning systems

base system protection and implementation	authentication
• network surveillance and software tests	• sophisticated authentication mechanisms
• proactive defence and information hiding	• prefer HTTP POST over GET
• encrypted data transmission only	• no persistent credentials
• proper interface specifications	availability
• restricted run-time environment	• suitable infrastructure
• privilege separation	• regular backups
• server-side processing of sensitive data	• distributed architecture
• strict (whitelist) configuration	• fall-back systems
	• load balancing

6.3 Implementation Concepts and Technical Remarks

In the bottom-most layer of Figure 1.1, technical aspects of base system security are in focus. This especially includes an appropriate infrastructure that does not provide a potential attacker with exhaustive information about internal processes and connections. Furthermore, implementation concepts, i.e., security deliberations realised during the programming process of base applications, need to be considered. As further subdivision of this layer and as structure for the following discussion, three areas will be distinguished: (1) base system protection and implementation for the very basic concepts of a reliable infrastructure, (2) the authentication process related to certain protocols, and (3) availability recommendations for ensuring business continuity even in case of smaller failures. As an overview, Table 6.1 collects all recommendations as discussed in the following subsections. Note that all recommendations are considered to be of almost equal value, such that no further statement will be expressed by the applied order.

6.3.1 Base System Protection and Implementation

This subsection primarily is focusing on recommendations related to protection on a very low layer. This is not usually specific for e-learning systems, but needs to be considered in practice for not allowing the circumvention of security mechanisms of an e-learning system by low level weaknesses. Recommendations deduced from the dependency of e-learning systems on base systems and infrastructural components are given as follows:

- *Network surveillance and software tests*
  In complex networks and system environments a lot of traffic is present and many different data packets are exchanged between components. To be able to detect fraudulent activities within the huge set of legal transactions, an evaluation of transmitted packets is necessary. Obviously, due to the vast amount of emerging events, recorded log entries, and packets transmitted over the network, a manual examination of such data for filtering legal from illegal activities or warnings is not efficient in practice (cf. Fig. A.39, A.67). Consequently, automated analysers for log files and packet analysers for current network activities are reasonable to detect fraudulent activities and to be able to intervene as soon as possible, e.g., with the help of automated intrusion detection and prevention systems. In particular, in case of successful attacks, data loss must be reduced and business continuity must be ensured to continue the provision of e-learning. Thus, appropriate reactions to attacks have to be deduced from logs. This includes respective modifications and adaptations to the security hole exploited for the attack, such that newly set up systems will not be susceptible again to the same issues. Besides this, deployed software needs to be examined concerning reliability, stability, and suitability for corresponding tasks. Also, exhaustive software tests

prior to a deployment in a productive environment are reasonable (cf. Fig. A.31, A.45), as well as an examination of the implementation and possible shortcomings.

- *Proactive defence and information hiding*
  In the fault tree as presented in Appendix A several possibilities for attackers are mentioned in order to gather information about possible target machines, e.g., network port scans (cf. Fig. A.28), detailed version information of server software (cf. Fig. A.33), exhaustive error messages with sensitive data (cf. Fig. A.34, A.37), or configuration data (cf. Fig. A.35). Such information can be very useful for preparing an actual attack, and therefore, access to these data should be prevented. In case of version information provided by software, this usually can be solved by reconfiguring corresponding software elements. Hiding network information is more difficult, since the fact of some port being open for connections cannot be hidden effectively without disturbing other services that need to connect to that machine. For this reason, Erickson [Eri03] proposes a proactive defence approach that does not try to hide information but rather render the given information useless. His approach gives a lot of fake information to an attacker, such that the real information vanishes within that vast amount of data. Hence, the gain of additional information is kept small.

- *Encrypted data transmission only*
  Concerning confidentiality and reliability of transmitted data, cryptography provides a sensible means by using encrypted data transmission only. A weak protection of the data transfer enables several attacks from simple eavesdropping to sophisticated data manipulation on-the-fly, i.e., by man-in-the-middle attacks (cf. Fig. A.58, A.61). With using standards like SSL (secure socket layer) or TLS (transport layer security) a server has to identify itself to the client, such that man-in-the-middle attacks can be prevented so far as users thoroughly check the received server certificate and deny connections in case of fake certificates. Despite the significant potential of (strong) encryption for increasing the security of data transmission, note that human fallibility can negatively influence this security gain. In case of web sites that are available over an encrypted HTTPS transmission, as well as over a plain text HTTP connection, users might only request the plain text version, since this usually is the default behaviour in today's web browsers. To take care of this problem, e-learning systems should only be available over an HTTPS connection and – for the sake of usability – redirect HTTP connections to the corresponding HTTPS version.

  Bear in mind that HTTPS stands for an HTTP connection on top of an additional encryption layer realising SSL/TLS, i.e., in case of only mentioning HTTP connections, e.g., for discussing protocol shortcomings, this does not affect the validity of this recommendation. Thus, respective discussions are not in contradiction to this demand for only using encrypted data transmissions.

- *Proper interface specifications*
  Functionality cannot arbitrarily be included by extension modules without increasing the risk of becoming vulnerable due to errors in such modules. But by accordingly crafted interface definitions taking care of data exchange controls, and a suitable implementation of security concepts within each module, this can be used for integrating additional functionality with still keeping a reasonable degree of security. Each module on its own can be made responsible for security critical activities and the base system can be informed in case of required or failed authentication. Furthermore, arbitrarily created roles in the base system cannot be entirely matched automatically with roles of different privilege levels as prepared in included modules (cf. Fig. A.25). But by providing a set of all available privileges to the base system, such privileges can dynamically be integrated into the central role configuration interface. Thus, administrators can still be provided with a single interface to assign privileges to corresponding roles. In addition, if interface specifications even include the definition of standard roles, module developers can be expected to prepare privilege settings for these roles.

- *Restricted run-time environment*
  The high probability of software to contain implementation errors, which could possibly lead to security holes, can significantly be mitigated by restricting the run-time environment. Approaches for doing so are described by terms like "sandbox", "jail", or "virtualisation". "Sandbox" here is used as general term for a secure separation of base system components, e.g., the operating system or a database system, and data directly related to the actual application. This means, in case of software flaws, errors can only be abused to access data within the sandbox (cf. Fig. A.37, A.48). Other external components are not threatened. Note that this concept only holds as far as the sandbox software is not vulnerable as well, and as far as the software does not result in a privilege escalation, with the result that an attacker may break out of the sandbox by abusing superuser privileges. Practical realisations of this concept are given by the "chroot" tool on Unix-like systems, where a virtual root directory gets created and software that is running in this "chroot jail" at most can go up to the virtual root represented by a subdirectory on the physical disk. A different approach for creating a sandbox is given by "virtualisation". This term refers to an abstract set of possible solutions describing the use of virtual machines as guests on a real server system, such that no process within the virtual machine can access elements outside this scope. Here even hardware access can be restricted to only take place over mediating systems.

- *Privilege separation*
  The term "privilege separation" usually refers to a dropping of superuser privileges if they are no longer needed by some process. This mostly can be found in network tools that provide a service listening to a so-called "well-known port", i.e., ports 0-1023. These ports may only be occupied by administrators, and therefore, software needs to be started with administrative privileges. Since after start-up special privileges are no longer required, they have to be dropped accordingly. Similar to these network services, the concept of privilege separation can also be transferred and applied to roles within an e-learning system. Thus, administrative privileges should only be available for corresponding persons in case they need them. For example, teachers in distant university scenarios might have assigned duties like entering results of assessments into user profiles, but their usual task within the system is mostly concerned with courses and learning material. Consequently, the rare but sensitive process of administering users can be protected by additional steps to temporarily increase privileges and by dropping these privileges after successfully finishing corresponding activities. Such a virtual change of roles with additional confirmations for respective tasks should be enforced to assure involuntary exploitation of privileges, and therefore, to mitigate human errors.

- *Server-side processing of sensitive data*
  With respect to Web 2.0 technologies and AJAX (asynchronous Javascript and XML) as modern concept for web applications, we have to consider locations of data storage and processing. By using AJAX, a lot of computational tasks can be transferred to the client machines, such that flexibility and performance of web applications can be improved. However, since trusted data storage and processing can only be ensured on server-side, this introduces additional security issues due to the increased complexity of distributed code. No process on client-side, although necessary for certain features, can be considered as trustworthy, and therefore, thorough validations of transmitted results need to take place. Otherwise, data could be altered and processes be manipulated (cf. Fig. A.23).

- *Strict (whitelist) configuration*
  Independent of the type of configuration that has to be set in a server system, whitelisting is usually preferable over a blacklisting approach. For example, in case of a web server that provides specific files for clients, a whitelisting approach has to ensure that only legitimate files will be delivered, e.g., files with suffixes like ".htm", ".php", or ".jsp". A typical

problem related to this recommendation is given by backup files made available on the server by mistake, e.g., files with a suffix like ".bak". Since backup files in case of only little changes are very similar to the corresponding protected files, this could result in a privacy and confidentiality leak (cf. Fig. A.10, A.37). Thus, all files not explicitly allowed to be delivered by the server have to be protected against unauthorised disclosure.

6.3.2 Authentication

One of the crucial processes in e-learning systems is given by authentication, since all subsequent security mechanisms base on a reliable identification of users. To allocate roles and privileges, users need to log-in using their personally assigned account data. For attackers, consequently, these data are valuable. In the following, the authentication basis will be considered, i.e., not the account data themselves, but technical aspects building a reliable base for the actual authentication process:

- *Sophisticated authentication mechanisms*
  Authentication processes that base on the exchange of specific data can be vulnerable to so-called replay attacks (cf. Fig. A.18). This means, an attacker can reconstruct and replay exchanged packets between client and server, and therefore, reproduces a successful authentication process even without knowing the exact authentication credentials. To mitigate this problem, authentication processes must include some sort of timestamps or unique (random) data only valid for the current session in order to render replay attacks useless. An approach basing on this concept is given by challenge-response mechanisms, such that users are confronted with a (randomly chosen) task as a challenge to be solved and then answer with the corresponding response to the challenge. This mechanism utilises the process for creating the answer to be the necessary secret, and therefore, the data exchanged between client and server could be eavesdropped without directly gaining useful information. Such mechanisms additionally can be realised by using smart cards processing challenge data on the integrated chip. Thus, this secret process for generating answers even can be protected against local attacks. However, the application of additional physical tokens or biometric authentication mechanisms might introduce requirements for hardware components like smart card or fingerprint readers leading to inconveniences with respect to limited mobility and flexibility (cf. Sec. 2.7). In case of distributed authentication mechanisms, such that an e-learning server does not manage the authentication process on its own, but relies on a distinct authentication service, the corresponding authentication protocol needs to be cryptographically secure and be elaborated for not providing attackers with possibilities to circumvent or disturb crucial data exchanges (cf. Fig. A.17).

- *Prefer HTTP* POST *over* GET
  During the learning process and organisational activities, perpetually data have to be exchanged between e-learning server and clients. For web-based systems, therefore, such data including request information must be transmitted using the protocol HTTP – preferably with additional encryption by an SSL/TLS layer. HTTP, in principle, offers two different possibilities for including and transmitting parameters: HTTP GET and HTTP POST. In the former case, all relevant data are integrated into the URL used for connecting to the server, while the latter case embeds parameter information in the message body and keeps the URL untouched. Concerning session hijacking or session fixation attacks (cf. I_3, Sec. 4.4.2 and Fig. A.18, A.22), the implementation should take care of these approaches and prefer the POST method over GET. Otherwise, attackers could inject URLs to users of the system, who by mistake or fallibility might activate such links, and parameters included in this URL are processed with the privileges of that client. Hence, to mitigate the risk of fraudulent parameter injection and involuntary execution by users, POST can be made the only acceptable way of receiving parameters.

- *No persistent credentials*
  Some vendors integrate so-called master passwords or default authentication data in their products to be able to reset this product in case of any mistake by a customer that harmed proper operability. But such credentials usually will be found by attackers (cf. Fig. A.19) either in case of (open) source code investigation or by cracking the password, e.g., by brute force. Note that since passwords usually never have to be accessed in plaintext, the storage in form of a cryptic string as result of a hash function, in general, is reasonable. Similar to the conceptual weakness of hard-coded master passwords are user passwords that are in use for a very long period of time. Such negligence significantly simplifies password cracking attempts. In case of frequently changed passwords, an attacker might have cracked a former password after some time, but due to already changed settings, this gain in information is no longer valuable and the attacker's effort was in vain. Thus, the integration of password lifetimes, i.e., enforcement for regularly changing passwords, is sensible to improve security.

6.3.3 Availability

In the following, the term "availability" will generally be used incorporating terms like "dependability" and "reliability" (cf. [And01]). This means, besides simply being available, corresponding systems also should provide sufficient resources and ensure reliable processing of data. Thus, in addition to trivial technical failures, this aspect of availability is strongly connected to denial of service attacks that might result in a lack of usable connections due to fraudulent manipulations or resource depletion (cf. Fig. A.54, A.56). Obviously, for successful e-learning, learners should be able to rely on the system, such that the probability of errors to occur is minimised, and that in case of an occurrence they can be fixed quickly. The following criteria should be considered for a practical implementation to ensure a high degree of availability, reduced probability of failure, and a fast return to a working system in case of emerging problems:

- *Suitable infrastructure*
  E-learning as a web-based application is confronted with the demand for a high level of interactivity to suffice modern learning theories (cf. Chap. 2). Thus, for promptly providing reactions to user input, the very basic requirements for successful e-learning are an assurance of sufficient bandwidth for the backbone network connection and suitable server systems capable of managing all requests of learners – even in times of peaks. E-learning systems most probably have to cope with increased amounts of requests near some deadlines, e.g., for assignments to be submitted, and therefore, bandwidth reserves need to be considered appropriately. Furthermore, since collaborative processes in certain scenarios are planned to be supported by the e-learning system in a way that all learners of the group can access data simultaneously, infrastructural components must be sufficient to enable efficient collaboration without increased delays or even data loss due to connection failures.

- *Regular backups*
  Besides technical availability of services, the business continuity in case of incidents or loss of data must be ensured. For this, relevant data need to be backed up regularly in such manner that this data storage and backup process is tested frequently to guarantee reliability. To not burden learners or teachers with this process, a central backup service is reasonable as duty for administrators. Note that personal data protected by access control mechanisms within the e-learning system could be backed up and afterwards be extracted from archives, such that this implies that only administrative staff should be allowed to backup the entire system's data. Similar to this, with respect to (concurrent) read and write access of several group participants in collaborative work, version management, history of changes, and locking mechanisms for critical activities are sensible for providing better reliability and for being able to undo unwanted alterations or to restore previous versions.

- *Distributed architecture*
  In order to minimise the impact of system failures, involved services like web server, database system, or authentication service can be placed on different physical systems to create a distributed architecture. In case of physical failures of one of these systems, the other ones are still available and do not need to be restored as well. Hence, for business continuity, only single services would need to be set up anew instead of the entire architecture, i.e., a so-called "single point of failure" could be avoided.

- *Fall-back systems*
  Similar to a distributed architecture is the demand for fall-back systems. In case of system failure in one of the services, or, more general, in a single position of the whole infrastructure, alternative systems can be provided to overtake the missing infrastructure element. This leads to a redundant set of components, where each service is provided by at least two physical systems. Short periods of time for switching between identical services guarantee a high degree of availability despite physical outage. Note that such redundancy also is necessary to solve the contradiction of keeping the system up-to-date with possibly requiring reboots and, on the other hand, being available almost all the time without outages reasoned by such update and reboot processes.

- *Load balancing*
  Regarding demands for a suitable infrastructure and fall-back systems, load balancing is desirable for actual infrastructure realisations. With this technology, the redundant, distributed architecture can be used more efficiently and less straining for single system parts, since incoming requests will be forwarded to the system currently least busy. Hence, huge amounts of simultaneous requests can be split in smaller amounts for each system providing the requested service, and therefore, probability of failure due to overload gets reduced. Thus, attacks like distributed denial of service (DDoS) can be easily mitigated to a certain extent by this additional concept.

6.4 Organisational Concept and Policies

Technical security as discussed above usually is considered as the most intuitive and primary field for implementing security means. This also is indicated by (official) security evaluation catalogues and methodologies like the Common Criteria (CC) that describe "administrative security measures" to be "somewhat peripheral to IT security" and "to be outside the scope of the CC" [ISO05, part I, p. 8]. Nevertheless, despite this decision, the relevance of administrative and organisational aspects is acknowledged:

> "However, it is recognised that a significant part of the security of a [target of evaluation (TOE)] can often be achieved through administrative measures such as organisational, personnel, physical, and procedural controls." [ISO05, part I, p. 8]

Threats as discussed in Chapter 4 and presented in Appendix A result in requirements for an appropriate security concept – of course, including organisational aspects.

For a transfer of tasks found in presence teaching processes, the following questions need to be considered and answered for a specific situation to fulfil requirements and find compromises between affiliated persons, such as teaching staff and learners:

- What data are necessary for supporting the learning process?
- Who is allowed to access these data? Can access to specific data result in negative consequences for learners?
- What activities should be supervised and are there any private areas without observation?

Table 6.2: Conceptual requirements concerning security services

data confidentiality	non-repudiation
• proper user management and authentication • roles and separation of duties • reduce data collection • front-end interfaces for hiding back-end information • pseudonymisation and anonymisation • transparency	• limit amount of authorised persons • record critical activities • public key infrastructure and digital signatures • trusted third party • digital watermarking
access control	**data integrity**
• default deny policy • proper transfer of roles to privileges • minimise functionality	• correctness assurance by digital signatures • concurrency control • consistency assurance • multiple confirmations for critical activities

In the following subsections, recommendations and requirements according to (not already in Section 6.3 regarded) security services are presented and discussed for a security concept with respect to limits as implied by educational and psychological aspects (cf. Chap. 2). As a collection of results, Table 6.2 provides an overview of recommendations as presented in the following. Note that the order of items has no further meaning due to almost comparable relevance.

6.4.1 Data Confidentiality

The collection of personal data for better personalisation and socialisation among learners is problematic. In informatics systems, it is quite easy to record activities of users and to create a set of data for each access or event. Nevertheless, an excessive data collection has to be seen critically due to the dual use character of logged events and supervision mechanisms (cf. Sec. 4.6.2). The fact of teachers taking notes in presence teaching, e.g., which pupil was missing and who participated in which manner during a course, is entirely different to informatics systems that objectively record all activities of users and do not miss any event only because of large amounts of participants. Subjectivity, fallibility, and obliviousness of teachers mitigate privacy concerns in presence teaching significantly. Hence, a simple and unchanged transfer of presence teaching approaches to e-learning scenarios most probably will not be accepted by (self-directed) learners. Thus, learning outcomes might be negatively influenced or learners will even refuse to use the system at all. In the following, recommendations will be discussed that take care of these contrary views on data collection and aim at a compromise that is acceptable for all parties:

- *Proper user management and authentication*
  Confidentiality in the sense of protecting against unauthorised disclosure of information is closely related to the authentication process and an appropriate user management. In case of stolen authentication credentials, attackers can easily pretend to be someone else and exploit gathered privileges. Therefore, prevention of account data disclosure is crucial for this service. First, revealing usernames by system messages should be prevented, i.e., protection against enumeration attacks (cf. Fig. A.20). This mostly results in demands for not giving more information than necessary in system messages concerning username creation, password reset, or failed log-in attempts. These messages must not give information about the existence of tested usernames. Second, entropy in generated usernames, (default) passwords, or session IDs must be kept as high as possible by sophisticated pseudo random

number generators. Low entropy could be exploited with respect to guessing such data (cf. Fig. A.19, A.20). Also, patterns in authentication credentials could be exploited to significantly reduce the search space for possible representation. Note that gathered information about some found accounts can possibly be transferred to other ones. Third, automated password cracking attempts must be hindered by demanding additional information not widely known, e.g., personal information like day of birth, or by demanding randomly chosen information to be shown as image, i.e., so-called "captchas". Furthermore, in case of failed log-in attempts, the system should include time delays to decrease the amount of passwords that can be tested by an attacker in a given period of time (cf. Fig. A.21). Restrictions on client-side, e.g., attempt counters using cookies, are not sufficient.

- *Roles and separation of duties*
  Role-based access control is commonly used and sensible in educational environments. To take care of adaptability demands concerning responsibilities and task sharing in certain courses, besides a global role assignment, the extension by local roles inside of courses gives much more flexibility. Global roles can be administered by the administrator for a global scope, while local roles could be made manageable for responsible course teachers to introduce further helper roles. However, as discussed in Section 4.6.1, roles with different scopes and inherent hierarchies within separated scopes can introduce several threats to be mitigated by an appropriate role management. Privileges and mutual influencing factors have to be considered appropriately to not allow this scope construct to be exploited to harm "subordinate" learners (cf. Fig. A.24, A.25, A.63). This means, while global roles may affect globally relevant data of learners and the system as a whole, local roles only may access locally relevant data, i.e., course specific progress and data connected to respective activities. Thus, a strict separation of duties is necessary. Although this requirement for small institutions can be regarded as less relevant, the concept of the system should consider this aspect for the sake of scalability to institutions of different sizes. Note that a separation of duties also affects data backups of the entire system and import/export functionality with e-learning standards (cf. Sec. 4.6.3 and Fig. A.48).

- *Reduce data collection*
  With respect to privacy protection, the national legislation of several countries demands to be as modest as possible concerning the collection and storage of personal data and related events. The less data are collected by the system, the less data need to be protected accordingly, and the less data are at risk to be uncovered by unauthorised persons. But, obviously, despite this basic principle of collecting as few data as possible, some sorts of data still are required to be stored. Thus, as a second protection level, the timely limit for data storage should be considered. For example, numbers concerning course properties like the amount of users online at a specific point in time or similar can lose their relevance if certain events have happened, e.g., when a course has been finished. Consequently, such data has no necessity of being stored any longer and can be deleted. In web-based e-learning systems, this also affects communication data. By using HTTP as transfer protocol, messages need to be stored on the server until all participants have requested these messages (cf. E_7, Sec. 4.3.3). Hence, for reducing the risk of data disclosure for these pieces of information, a regular clean-up process is desired. This process should sign out users that have never accessed a course or were inactive over certain periods of time. In addition, it should delete old log files and take care of module specific clean-up processes. For communication facilities, old messages should be deleted or archived according to the configuration.

- *Front-end interfaces for hiding back-end information*
  Besides input filters to sanitise received data, the output of internal data can also be filtered or processed to prevent disclosure of sensitive contents. Thus, users can be provided with front-end interfaces to manage certain processes, while the actual processing takes place in the back-end with keeping relevant data hidden from users. Such front-end services and in-

terfaces, for example, could be given by internal message systems that use e-mail addresses without the need for revealing them to respective e-mail senders (cf. Fig. A.8). Consequently, messages can be typed in the e-learning system and be sent by this application even without showing actual contact information. Clearly, this approach can be transferred to other e-learning components in similar manner. For example, automated statistical reports can be given with assuring anonymous output and still hiding detailed log data from users.

- *Pseudonymisation and anonymisation*
  A lot of data do not lose explanatory power if not related to a specific person, e.g., quantitative access statistics. Hence, if possible, such data should be stored anonymised or at least pseudonymised, such that these data cannot be exploited to spy on users in the system (cf. Fig. A.8, A.16). If anonymisation of data could become counter-productive in certain scenarios, then delivery and presentation of (non-anonymised) data after certain correlation processes should be in an anonymised way as far as possible for the respective situation. For example, this affects the logging of critical activities, where user identification is necessary, and the collection of empirical data requiring further details about related users to estimate reliability of data entries. With this double-sided concept, personal data is available in the back-end system, while only non-critical data are delivered to the front-end for a certain set of persons.
- *Transparency*
  With respect to user acceptance, the transparency aspect is crucial. Users mostly will accept exhaustive surveillance, if they at least are informed about it, and if this surveillance can be reasoned in any way. But in case of accidentally detecting surveillance without being informed in advance, resistance will be the consequence (cf. Sec. 4.6.2). Hence, transparency in this case means information about the degree of recording activities in the system and about general conditions like durations of storage or application of anonymisation. With respect to users' influence on the usage of their data, some national data protection legislation demand a confirmation to data collection and an agreement to the usage of explicitly collected data. This has to take place in a way that cannot be overseen and accepted by default, e.g., using a checkbox that is checked by default in an online form. To simplify data processing, a form can be stated in a general way for letting the user accept all related data collections and processes affecting these data in advance. Thus, only a single form prior to connecting to some services is sufficient instead of several confirmations during the activities. In addition to system related data collections, handshake protocols are sensible for requiring mutual confirmations concerning state changes, and therefore, data publication for certain users. This especially concerns course enrolments of users initiated by course teachers in corresponding scenarios (cf. Sec. 4.6.1 and Fig. A.63). Additionally, the confirmation demand supports the protection of data for not being collectible without informing affected users.

6.4.2 Data Integrity

Integrity, correctness, and consistency of data are of value, but quite difficult to guarantee. In presence teaching with human teachers and only paper-based notes, unintentional modifications of notes can almost completely be excluded. Even unauthorised modification of own notepad entries are less likely or at least can be detected in most cases due to a different handwriting. In case of e-learning with centralised data storage and digital notes only, this must be considered in a different way. Besides access control mechanisms based on roles as introduced in Section 3.2, this primarily can be summarised to the following recommendations:

- *Correctness assurance by digital signatures*
  For spending time on learning certain content, learners should be in the position to rely on its correctness. Reliability in this case can be assured by letting these contents be checked

by experts who finally digitally sign all related data elements. With this, manipulations and transfer errors can be detected in an early stage due to falsified digital signatures. Correctness besides learning contents also affects communication and cooperation messages (cf. Fig. A.14), such that the use of digital signatures should be considered in a system-wide manner to not neglect any important data set. Note that contents can even be signed by several experts to signalise common agreement.

- *Concurrency control*
  Collaboration together with communication and joint work on solutions are crucial components of e-learning systems. But they also introduce problems in case of concurrently working on the same documents over the network (cf. Fig. A.12). If related functionality is implemented regardless of possible conflicts, then race conditions are likely to happen or even can be provoked by attackers (cf. Fig. A.13). Although race conditions are restrained to very small periods of time, and as such, often are considered to be very improbable, an e-learning system has to take care that no learner intentionally or unintentionally can delete the work of others due to concurrent access to the same files. Thus, since race conditions actually can occur and can be of great impact due to underestimating their risk, blocking mechanisms like semaphores are sensible and should be implemented for this reason. In addition, mechanisms taking care of latencies and timestamps for almost simultaneous modifications of same elements are reasonable to inform all learners when someone else is currently modifying the same pieces of data.
- *Consistency assurance*
  Data consistency is essential for productive work. In case of slightest inconsistencies, this will perpetually get more and more problematic, since other processes might rely on wrong data, which again leads to wrong results and new wrong data. Hence, consistency control is an important part of ensuring data integrity in the sense of correct data processing. Principles as known from database management systems like transactions with their all-or-nothing approach are sensible, as well as file access locks can help to prevent race conditions in case of critical activities (cf. Fig. A.13).
- *Multiple confirmations for critical activities*
  In web-based scenarios, states like a successful authentication are usually managed in sessions. These sessions are stored on the server, while the client receives a unique and randomly chosen session ID to identify himself and to let the server find the corresponding data sets again. Thus, if an attacker is able to reproduce or steal this session ID, he can act with the same states as the legitimate user could do, i.e., he also is authenticated to the system and gets full access with the privileges of the user. Since several attacks aim at stealing such session IDs (cf. Fig. A.22), security is aspired by demanding additional authentication steps and confirmations. Thus, even if an attacker could hijack a session without knowing the authentication credentials, critical activities are not possible. Also, sensitive data may not be changed, as far as the system again requires the user to be authenticated.

6.4.3 Non-repudiation

Certain activities in e-learning systems require accountability and non-repudiation to be implemented. This especially holds for activities that are influencing grading processes or obligatory tasks for being issued a certificate, e.g., submission of assignments and collaborative work. In these scenarios, no affiliated party should be in the position to deny successful submission or to accuse the other party for having altered original submissions. Hence, integrity protection for the submission is one part that needs to be implemented, but also authenticity, i.e., who actually was involved in some activity. The following recommendations support these aims:

- *Limit amount of authorised persons*
  The most trivial solution to the problem of who may have changed certain configurations or data entries is given by reducing the amount of users having access to these data (cf.

Fig. A.57). If only few persons have access, then these persons can be controlled more easily than a vast amount of users with the same privileges, i.e., horizontal attacks are less likely (cf. Fig. A.26).

- *Record critical activities*
  The easiest approach, and in most e-learning systems an already integrated functionality, is a logging of all relevant activities. Every critical action in the system triggers an event that can be recorded in log files and afterwards be evaluated by administrators in case of emerging problems. Note that for the sake of security, log files should not (only) be placed on the same system as the corresponding program logic receiving and recording respective events. Ideally, event details securely get transmitted to an external append-only system where no manipulation is possible by only using the provided interfaces (cf. Fig. A.38).

- *Public key infrastructure and digital signatures*
  For assuring authenticity of actions, mechanisms beyond authentication and access controls are necessary. For critical actions and submissions, therefore, digital signatures can be demanded to ensure integrity of submission and authenticity of the signing party. Thus, transmitted data – at least for critical processes – can be signed by corresponding authors, such that unauthorised changes can easily be recognised by receivers due to wrong signatures. Note that in case of unsigned contents, these data would have to be discarded for not circumventing this additional security means. To equip every learner with a required public-private-key pair and for taking care of identities of learners, a public key infrastructure (PKI) can be built up at corresponding institutions, such that administrative staff or executives act as registration and certification authorities in this PKI. This results in keys that are checked to belong to the pretended user as far as this key did not get lost or was shared among a group of learners. However, liability in case of sharing keys can be transferred to the users illegally distributing their secret keys.

- *Trusted third party*
  For mutual accountability, where the submitting and receiving parties must not be able to deny certain activities or reception of data, trusted third parties can be used (cf. [GRV05]). Such third parties can act as interim receiver of data, or only be used as cryptographic mediator that checks signatures and hash values to guarantee authenticity and integrity of submissions. With a trusted third party as interim receiver, the actually meant receiver can several times request the data from this party, such that he cannot claim to never have received respective data from the designated sender. In the other direction, the actual sender has to provide the data until the trusted party finally confirms successful reception. As a special case for accountability, private authentication can be considered [Aba03], i.e., authentication while still acting anonymous within the e-learning system (cf. Fig. A.16). Certain activities like the evaluation of courses may only be conducted once, and therefore, access has to be logged. Here, trusted third parties are sensible to overtake an authentication process without delivering identity information to the main e-learning system. This means, authentication and the actual working process within an activity are not on the same physical system, but strictly separated. These servers do not influence each other and enable such scenarios by an appropriate exchange of (cryptographically secure) messages.

- *Digital watermarking*
  Concerning copyright and reservations related to unauthorised distribution, mechanisms are desired to protect against such illegal activities (cf. Fig. A.10). Especially with respect to the effort for creating good learning material, an appropriate protection gains in importance for authors and teachers. Thus, policy enforcers and embedded viewers without directly making material available for learners, as well as digital rights management or other copy protection mechanisms are conceivable. Note that all these mechanisms rely on the used software and possibly can be bypassed by accessing those contents with other applications. A solution basing on cryptographic mechanisms storing all data in encrypted way even

on the client machines is presented in [Gra03, pp. 123-126] under the term "encryption enforcement". Since this mechanism demands to be online for retrieving session keys to decrypt contents for every single access, it is not handy enough for providing flexible and fast e-learning solutions. Another approach not focusing on preventing copying and distribution, but tracing illegal copies back to the last authorised accessing person, is given by digital watermarking. Media can be marked in a way that learners only get copies that invisibly contain their personal identification code in an encrypted way [Dit00]. These watermarks are robust enough to even endure transcoding to different file types. Hence, although prevention is not possible with this approach, liable persons can be identified and punished accordingly.

6.4.4 Access Control

Security basically relies on an appropriate access control mechanism and corresponding privilege settings (cf. Fig. A.24). To support such configurations, conceptual deliberations are reasonable for reducing probability of wrong settings. In the following, recommendations that aim at simplifying privilege management and at reducing error-proneness due to weak default configurations will be introduced and discussed:

- *Default deny policy*
  Similar to a whitelisting approach for file access, privileges should follow a default deny policy. This means, only activities that are explicitly allowed by corresponding configuration settings may be conducted by users. Other activities have to be denied. Thus, even in case of improper administration (cf. Fig. A.27), where several possible activities in the system have not been assigned privilege settings and rules, this will not result in security breaches. Note that such a policy especially is of relevance when considering dynamic module inclusion with own privileges and activities that are not known to the core system prior to an actual inclusion. Consequently, a general consideration of these particular privileges is not possible. Without an appropriate configuration, this also could lead to a significant impact on security if preferring a default accept policy.
- *Proper transfer of roles to privileges*
  The actual personnel structure within an educational institution should be easily transferable to the access control mechanisms to support users in configuring the system accordingly (cf. Fig. A.25). If these two concepts, i.e., access control scheme and personnel structure, are too different, then no sufficient security can be achieved without the necessity of using several compromises either against security or single persons. Furthermore, due to the fact of persons acting individually in certain situations and not following general group duties or habits, possibilities for adjusting privileges on a per user base can be helpful.
- *Minimise functionality*
  A common problem of privilege settings is given by implemented functionality that has to be aligned to available privilege options. If only a single privilege option is connected to functionality that consists of several sub-functions and activities, i.e., the privilege system is too coarse, then users possibly get assigned privileges for sub-functions that are not meant to be used by those persons (cf. Fig. A.25). Thus, functionality behind single permissions should be minimised and the privilege management be as fine-grained as possible to enable a configuration that actually fits the required personnel structure.

6.5 User Support and Human Factor

Besides the general research for software ergonomics and user interface design (cf. [Shn92]), the connection of security and usability became a focus of research. Yee stated for the relevance of usability that

Table 6.3: Recommendations concerning user support and human factor

guidance and (security) usability	security education and user involvement
• reduce secondary interaction	• security education
• plausibility checks and fall-back settings	• provide documentation and manuals
• no error correction in sensitive areas	• policies and enforcement
• provide guidance	• concept audit
• confirmations	
• adapt complexity	

> “[. . .] correct use of software is just as important as correctness of the software itself” [Yee02, p. 278].

Therefore, activities and mistakes due to user fallibility need to be considered in more detail. In particular, user education, in relation to intentional and unintentional activities, will be in focus. Findings from a psychological perspective are presented by West [Wes08] who investigated user behaviour concerning security within informatics systems. He points out aspects that describe the basic problem very well, e.g., “Users aren’t stupid, they’re unmotivated”, “Safety is an abstract concept”, or “Security as a secondary task”. Especially the latter claim indicates contrary views on informatics systems. On the one hand, users want to follow their primary tasks, i.e., use the computer as tool for finishing some work. On the other hand, security people try to make computers more secure and prevent data loss or manipulation, such that these security mechanisms might interfere and even hinder the primary task. Tim Mullen stated this conflict as follows:

> “We spend money, increase administration, and take away functionality. Is it any wonder that security people are so misunderstood?” [Mul03]

Thus, to bring security deliberations together with expectations and needs of users, some requirements as presented in the following have to be fulfilled. Table 6.3 summarises recommendations to be presented in the following subsections. Note that the order of recommendations does not correlate to their specific relevance and impact. All are considered to be of comparable relevance.

6.5.1 Guidance and (Security) Usability

As stated before, users are mainly focusing on their primary task within the system and usually do not want to be involved in administrative tasks related to security. But since computers on their own cannot make decisions with sufficient accuracy in certain critical situations, it is sometimes not avoidable to inform users about critical events and to ask for interaction. However, the research field of security usability examines ways of making security relevant decisions as easy and understandable as possible for inexperienced users (cf. [HS07]). In the following, recommendations will be given that take care of the primary task of users to avoid unnecessary user interaction. Nevertheless, users should be involved in decisions with proper guidance and support for not discouraging and overstraining them:

- *Reduce secondary interaction*
  In fact, security mostly is not in the focus of users – it is at most of secondary interest. Hence, in case of security breaches and warnings popping up, the user tends to make quick decisions to continue with the primary task. This means to shift the decision making process as far as possible to automated routines following a reasonable set of rules not threatening the system more than tolerable. A possible realisation, for example, is given by an automated security agent as presented in Section 6.6.

- *Plausibility checks and fall-back settings*
  Complex informatics systems usually provide many configuration options to enable users to adapt the system to very specific situations and needs. For the sake of usability and user support, an e-learning system should implement plausibility checks for configurations. Thus, in case of contradictory entries or critical adjustments, warnings will be displayed and confirmations will be necessary. This can prevent loss of availability of a system because of incorrect and damaging configuration activities. Furthermore, a possible solution for not overstraining less experienced users and for simultaneously providing sufficient adaptability for expert users is given by a level-based structure of configuration options. This means, a base set of simple options can be presented to beginners, while an expert level can be chosen by experienced users to make other options available. However, in case of contradictory or wrong configurations by users (cf. Fig. A.26), the system can fall back to secure default settings. But note that this must not be implemented for security critical options, since otherwise this might be exploitable similar to error corrections.

- *No error correction in sensitive areas*
  Since the term "hacking" usually is defined exactly in the way of using some service against its specifications (cf. [Eri03]), automated corrections of input can enormously simplify attack attempts. Error correction especially in security critical areas could help intruders in conducting attacks, since they are allowed to work less precisely (cf. Fig. A.42). Hence, usability and security are sort of contradictory in this context. Mentioned security critical areas, therefore, clearly have to be defined to enable error correction and fault tolerance in remaining uncritical areas for the sake of usability.

- *Provide guidance*
  Not every situation emerging in practice can be anticipated, and therefore, user interaction cannot be prevented in all cases. For supporting users in making sensible decisions where asked by the system, they need to get an impression of what is going on inside and what the consequences in terms of gains and losses will be for each possible decision. Inexperienced users get overstrained very quickly, if too many technical details are mentioned in messages and the language is chosen for experts. Therefore, users usually do not read system messages reliably, even if contents are of importance. A possible guidance for required user interaction is the use of colours to indicate importance and possible impact on the system. In addition, text messages should be kept short to allow users to see all contents within a few seconds. Further information for the decision making process can be offered by buttons that redirect users to respective information.

- *Confirmations*
  Integrity of data is mostly endangered by unauthorised and unintentional alteration of contents. Both cases can be solved with including agreement and confirmation demands as far as possible, such that in case of critical actions on personal data, e.g., course enrolment or changes in booked services, the corresponding person needs to agree or at least confirm reception of notifications (cf. Fig. A.63). In case of unintentional activity, e.g., publishing of private data (cf. Fig. A.8), confirmation dialogues can be used to safely abort unwanted actions. Furthermore, additional dialogues to be confirmed may even emphasise the criticality of certain activities, such that users might think again, whether this activity is actually wanted.

- *Adapt complexity*
  Due to human habits of choosing the way of least resistance, systems should be programmed in a way, that the most secure path is the easiest while relaxing security policies should require increased efforts to be finalised. Especially due to the fact that "users do not think they are at risk" [Wes08, p. 37], it is necessary to not make it too simple to work with excessive privileges and to change security relevant preferences in the system. Thus, findings of (security) usability can be applied to simplify understanding of security relevant activities.

On the other hand, security critical activities can be made even more difficult. To change critical things, a user really should need to know what he is doing there and what the consequences could be. This aims at preventing users from choosing bad security adjustments. Note that this also is related to privilege separation, i.e., only temporarily increase privileges and drop administrative privileges again if no longer needed (cf. Sec. 6.3.1).

6.5.2 Security Education and User Involvement

The primary goal of this thesis is the investigation of security mechanisms for e-learning systems while maintaining user acceptance. With focus on users of the system, this can be managed by an adequate security education and by user involvement. First, security education for users is desirable (cf. [GI00]) to foster awareness of implemented security mechanisms and their respective goals. Users of the system can only do the right thing and act correctly if they get informed appropriately of what the right thing is. Although security education introduces further requirements and demands for additional staff and appropriate course materials, a lack of education may result in considerably more disadvantages. If security education would be neglected, users might be willing to circumvent security measures as far as possible, since the meaning of respective measures is unknown in the same way as possible negative consequences of bypassing mechanisms. Second, user involvement is an important aspect of creating an appropriate and widely acceptable concept. In Section 2.7 limits for security mechanisms were stated with respect to privacy concerns and surveillance within e-learning systems. This aspect is one of the most critical and specific parts of e-learning. Depending on the target group, teaching staff should be enabled to access more or less detailed information about the learning progress of learners and emerging problems while concentrating on learning contents. Thus, all groups of users (including learners) should have possibilities to take part in the decision making process, or at least in the evaluation of underlying concepts to avoid later problems concerning unwanted situations protected by corresponding concepts.

In the following, recommendations concerning user involvement and their education for better understanding security concepts are presented:

- *Security education*
  The concept of security is hard to understand for non-technical persons, since an exploitation of basic systems and protocols seems very abstract and users do not feel to be at risk [Wes08]. Security is a concept that usually only gets visible in case of its absence, such that incidents can occur. Consequently, fostering comprehension of security means and the necessity of obeying security policies is important. Users at least should be educated concerning their personal relevance within the security concept, e.g., attacks against human users for gathering sensitive information (cf. social engineering [MS03], Fig. A.7). Note that system messages also can provide attackers with valuable information. Thus, either usability with exhaustive information about occurred errors or concealment of information for not supporting attackers can be realised. In fact, to follow the latter approach, users need to have an impression of internal workflows, i.e., appropriate education of users is required to let them track general error message back to possible reasons.
- *Provide documentations and manuals*
  To support correct use of software, users must be appropriately informed about correct usage and consequences of doing wrong. In the Common Criteria version 2.3 [ISO05] "operational user guidance" is explicitly regarded in assurance class AGD ("Guidance Documents"), family AGD_USR [ISO05, part III, p. 141]. This assurance class takes care of guiding users to avoid human errors due to misinformation and mistakable presentation of security relevant aspects. Thus, this means to provide users with proper manuals of how to conduct certain activities and with documentations containing information about connected systems and relevant system internal data (cf. Fig. A.15). Due to changing users

(in a long-term perspective) and possibly unintuitive organisation forms, obviously not all activities and requirements for processing certain tasks can be known to learners. This also includes information about important contact persons, e.g., for a notification in case of security incidents. Therefore, in general, reliable and correct manuals should be available to describe activities in understandable and comprehensible manner. This can provide support and decrease frustration. Of course, such informative and regulative material needs to be available and correct to be useful.

- *Policies and enforcement*
  Besides informal documentations and manuals, formal policies are sensible to demand specific behaviour by users, e.g., minimum password lengths, maximum duration of inactivity during a session, or maximum number of simultaneous connections by using the same authentication data. As far as such policies can be formalised, the e-learning system can take care of their application by pervasively checking user activities and preventing incorrect behaviour with respect to stated policies, i.e., policy enforcement (cf. Fig. A.15).
- *Concept audit*
  Conceptual aspects, like an insufficient separation of duties or incomplete access control schemes, can lead to an enormous negative impact on the success of e-learning systems. Target group and acceptance evaluation must be considered together with balancing loss of private data and possibly more complicated administration (cf. Sec. 2.7). Hence, pedagogical and psychological aspects (cf. [Gud01, KHS05]) are relevant for the concept of e-learning systems. To achieve a concept that is acceptable for all affiliated roles, audits may be established prior to a productive usage and, in addition, regularly for already deployed systems. These audits should be processed by a team consisting of representatives of respective roles, i.e., not only technical staff should take part, but also learners who, in the end, will be affected by implemented concepts.

6.6 Security Agent for Practical Support

Security mechanisms often need deeper understanding of underlying structures and protocols to be used on the network layer. This cannot be demanded from all learners. Especially in case of complicated and time-consuming activities for security assurance, an automation is sensible to not distract learners more than necessary from their learning process (cf. Chap. 2). For this, in the presented research project, an open-source proxy server was developed by a students' project group (Oct. 2006–May 2009) under guidance and supervision of the author [ESS07]. The final product can be downloaded at URL `http://www.die.informatik.uni-siegen.de/pgproxy`. Since a proxy server logically runs on ISO/OSI (International Organization for Standardization / Open Systems Interconnection) layer 7, the application layer (cf. [ISO94]), it is able to examine and alter packets just like a human user could do. Thus, this proxy server as security agent primarily aims at being sort of a personal secretary for learners, i.e., it can overtake certain security-related tasks from the learners to keep them concentrated on their learning process. As proof of concept and first security task, the proxy server was extended by a module to sign communication contents sent to a learning management system (cf. Fig. 6.2). In the following, first, general aspects of this security agent concept will be introduced and discussed in Section 6.6.1, and second, the signature functionality will be presented in more detail in Section 6.6.2.

6.6.1 General Specifications

The implementation aims at a small and fast program that is available for most platforms, i.e., a platform independent implementation is desired for not excluding possible learners from using this agent. Besides this, the security agent will have to cope with the following tasks:

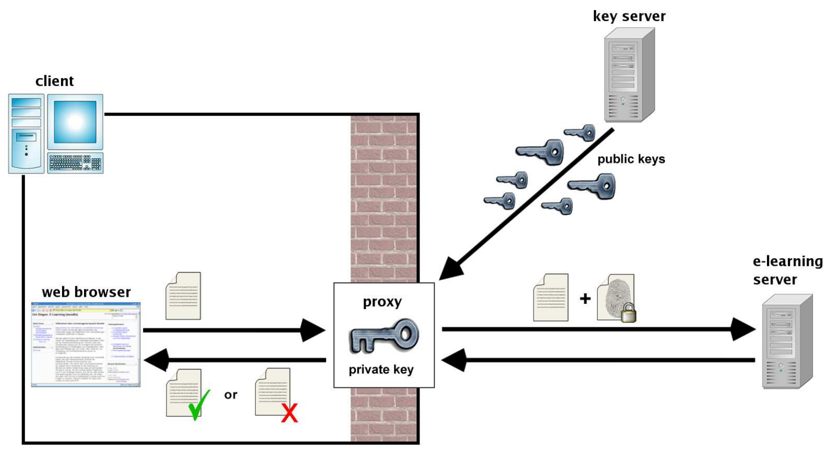


Figure 6.2: E-learning architecture with signing proxy

1. network adjustability,
2. task specific operations,
3. interaction with user, and
4. exclusion prevention.

1. Network adjustability: With respect to a proper operation of the network, the proxy component of the security agent, of course, has to be fully compliant to the HTTP protocol (cf. [FGM$^+$99]) and to be able to cope with several simultaneous connections. That means, every connection established through the proxy server needs its own thread. Otherwise, the first connection would block the proxy server and no further connection could take place before the first one has been reset. For the sake of performance of connections, we propose the restricted storage of packets by forwarding data as fast as possible to the LMS. If storing the whole message before forwarding data to the meant target, i.e., the e-learning server, this can get very time-consuming when considering the transmission of large file attachments. The signature still can be computed on a stored copy of the whole message and be attached to the end of the stream. For being able to use the security agent in the wide variety of existing networks, it has to be highly configurable. For example, if the network configuration is very restrictive by using particular firewall rules or demanding other proxy servers in the network, then our proxy server must be able to run on different ports of the client as well as supporting chains of proxy servers to cooperate with demanded other ones.

2. Task specific operations: The concept of using a proxy server as secretary is very flexible. In the sense of a security agent doing tasks instead of the user, different jobs are conceivable like signing contents, encrypting contents, control routing, and others. Besides the digital signing task as presented in more detail in Section 6.6.2, especially the authentication process could be automated. As long as the passphrase for a private key of the user is strong and kept secret, a digital signature can be used as the authentication method in challenge-response scenarios where the server sends a randomly generated string and demands an encrypted version of this string as answer. This enables a cryptographically secure authentication method that cannot be easily repeated by an eavesdropper. Furthermore, single sign-on approaches can be realised, such that the user only has to authenticate to the security agent and not to the LMS itself. Hence, even automated log-outs because of long idle time on server-side would not distract learners with an automated re-log-in. Obviously, all possible tasks require different adaptations to passing packets. Consequently, this second task cannot be specified uniformly, but only interfaces can be given to enable extensions by modules and libraries implementing relevant steps and processes to fulfil requirements for successfully realising a corresponding task.

3. Interaction with user: Although the integration of a security agent primarily is meant to reduce technical user interaction for keeping learners concentrated on learning, a certain amount of interaction still cannot be avoided, e.g., to inform users adequately about task specific results. Since the security agent has to interact in some scenarios, such as warning the user if a security task led to errors or if security problems were discovered, this should preferably take place in the user's language. This will allow them to be comfortable and as least distracted as possible. Consequently, the agent should be dynamically adjustable to the language of the host system. For system messages, different approaches are possible, e.g., providing a graphical user interface or modifying the HTTP stream. In the latter approach, the security agent can add some messages or a colour bar, for example, on top of the requested web page to signalise the security status of this site. Additional possibilities for integrating more decent status messages are given by CSS where contents can be arbitrarily placed on the delivered web page. Even translucency would be supported with CSS in recent browsers.

4. Exclusion prevention: Since security improvements, by using the proxy server as security agent, demand modifications and evaluation of data packets, the proxy must be able to compute received data. This means, if client and server are trying to negotiate an encrypted connection or a compression that is not supported by the proxy server (cf. Sec. 6.6.3), then this would lead to an exclusion from communication. An example is given with the use of encrypted connections based on SSL or TLS. Since such end-to-end encryptions take place between client, i.e., web browser, and web server, the proxy server will only be confronted with an encrypted HTTP stream. For being able to cope with such a situation, the proxy server has to adopt the client part for this encryption. Thus, the proxy server negotiates encryption parameters with the server, and the web browser sends its data either unencrypted to the (local) proxy server or encrypted by establishing a separate SSL/TLS connection between browser and proxy. Of course, the proxy server has to inform the user about server certificates used for the SSL/TLS encrypted connection. The proxy server must not be the deciding part, whether a certificate is trustworthy or not! But with information about most common certificate authorities it can support the user in his decision by verifying the server certificate in advance.

6.6.2 Integrity Protection by Digital Signatures

Since the first prototype of this security agent was implemented for the specific task of integrity protection by digital signatures, the following explanations will focus on this particular example to get more concrete.

Considering available LMS, the assurance of integrity, e.g., of learning content, as well as parts of non-repudiation, e.g., for proving the participation of some student, is not implemented satisfactorily. Digital signatures give an appropriate means for realising such concepts – they even allow multiple signing of learning content to signalise consensus among teaching staff. Yet although digital signatures have a mainly accepted value for security, the application of such methods demands a lot of interaction with users. Considering the kinds of messages sent to an e-learning system, i.e., learning content created by teaching staff or communication messages in forums and chats (by tutors and learners), it becomes obvious that especially in case of very short messages (like found in chats), the manual application of digital signatures would mean a lot of inconveniences that will enormously interrupt the learning process and disrupt the communication with other learners. The introduced security agent concept will be used to cope with these aspects by an automated application and verification of signatures.

For signing and verifying contents exchanged with an LMS, specific tasks can be given as follows, referring to the numbering as introduced above:

2a. LMS recognition: Since there are a lot of different LMS available, the extensibility of the security agent is highly desired. Request patterns for LMS functions can be roughly described by the tuple `(host,path)`. The first element describes the host URL for the LMS, while the second element describes the relative path on the target machine. For being able to recognise whether a user tries to connect to a known LMS, the agent has to compare data like host URL and requested site from the received packet against patterns stored in its configuration files. If none of the patterns match to the currently requested page, the proxy server, of course, must not alter any detail of the request and has to forward packets without any further delay, i.e., the network functionality must not be reduced more than absolutely necessary. Note that proxy tasks 2b, 3 and 4 may only be applied in case of communicating with a known LMS, otherwise the communication would (at least slightly) be disturbed by unnecessary computation of received packets without giving any advantage. In case of exchanging data with a web server not known to host an LMS, this results in a transparent proxy server forwarding messages conform to the HTTP specification [FGM^{+}99]. Thus, the user is not confronted with the proxy at all.

2b. LMS specific signing and verification: Signing contents and the verification of embedded signatures, of course, are two different subtasks. Signing messages sent to an LMS requires information about variable names contained in packets sent to a specific site on the LMS. The agent needs such information in order to separate contents to be signed within the observed HTTP stream from data that can be ignored. In addition, the security agent needs to know how to combine these variables to a new simple message that finally can be digitally signed. A list of variables appears sensible to mention the relevant variables in an ordered way. For actually being able to sign extracted variables, the agent first has to gain access to the private key of the logged in user. This key can be stored on a smart card, a USB stick, or, if the pair of keys was created on the currently used computer, locally on the harddisk drive. Since private keys should be protected by a passphrase or other authentication methods, the agent will have to ask for an identification of the user. When the key can successfully be decrypted, it needs to be stored in main memory to give the agent the possibility to use it several times without asking for authentication again at every access. Note that, after quitting the security agent, the private key stored in main memory should be scrambled, such that no other users, including local administrators who get logged in later, can restore the plain private key by analysing main memory [SS99].

For the other direction, i.e., incoming messages, the security agent acts as control instance for contained digital signatures. Note that the verification process needs a lot more descriptive information than the signing process. Variables sent to the server can only be requested again in the form of a flat HTML page without forms or other dividing structures. The messages as presented by the LMS are no longer stored in variables, but are embedded in HTML page text. In the configuration files, data about the recognition of digital signatures embedded in these pages and the possible segmentation in single message blocks need to be given. To actually evaluate embedded signatures, corresponding public keys of signers must be known to the agent. For this purpose, a keyserver can be used that provides all known public keys, e.g., within an institution's PKI.

6.6.3 Realisation

A main realisation goal of the security agent in general is the conformity to the protocol HTTP in version 1.1 (HTTP/1.1) as specified in Request For Comments (RFC) 2616 [FGM^{+}99]. Thus, the agent has to fulfil all requirements concerning the proxy functionality itself before a particular security functionality may be integrated. In the students' project, a rapid prototyping approach was followed for this goal to be able to thoroughly test the proxy server with different web servers and hosts. This especially turned out to be advantageous, since several web servers on the Internet act according to rather deprecated, but still allowed protocol specialities as described in RFC 2616 [FGM+99]. Since all protocol details like different usages of address requests and their handling are specified in this standard, they will not be further discussed in the following.

Second, the proxy server has to be able to identify relevant parts of the HTTP stream and to react appropriately. A main problem concerning the ability of recognising relevant parts in the stream is that the stream itself must be kept readable for the proxy server. HTTP/1.1 allows several ways of encoding and compressing data. Usually a web browser informs the web server about its supported compression types. The web server, on the other hand, compares this information to its own supported formats and decides accordingly. If the proxy server in the middle does not support the negotiated compression, then it will not be able to work with received data. Hence, the offerings of the web browser need to be considered and possibly be adjusted to the capabilities of the proxy server. Of course, such adjustments are only necessary and sensible if the connection does match a pattern as introduced in the former section, i.e., a connection to a page or function within a known LMS is requested (cf. step 2a, Sec. 6.6.2).

With respect to the signing task, all relevant variables need to be extracted and concatenated to sign them all at once. Note that the list of variables and their order must not be modified, if it already has been used in the given way by any client. Otherwise, the modification of the variable order or the number of variables would lead to a different plain text, and therefore, will lead to another signature. Hence, old signatures would be falsified anytime they get checked with the new verification method. Since HTTP supports different request methods affecting the composition of messages, we have to distinguish their handling. The most commonly used methods are POST and GET. The proxy server has to be able to extract the variable names and contents of all these kinds of messages. While in GET messages variables are coded into the URL, in POST messages the URL does not contain any variable, but the packet body does instead. This means, for POST requests, so-called "multipart/form-data" messages are sent, where every variable gets its own field within the packet body. These fields are separated by a (randomly generated) boundary string introduced in the packet header. Hence, before processing received data, the type of message composition has to be determined.

To manage transparency for all users, it is sensible to store the signature in a way that it cannot be seen at first sight by any learner not using the described proxy server. For achieving this, the digital signature has to be stored in HTML tags that are not displayed. Since most LMS contain a content management system that filters user entries, there is no general solution how to keep signatures hidden. In the following section, emerging problems when applying this security agent to Moodle will be discussed.

6.6.4 Application in Moodle

Moodle is used at the author's university, such that experiences could be collected concerning its practical usage, implementation, and security concept. Consequently, for a prototype of the introduced security agent we focused on supporting this LMS. Moodle uses specific scripts for each learning activity with appropriate parameters like a unique course identifier to specify the actual course or context. Hence, only a finite number of sites representing such activities must be observed, and therefore, the tuples representing search patterns can easily be created. For the verification of embedded signatures, corresponding HTML pages have to be parsed. Forum entries, for example, have hierarchical structures to separate main statements from their corresponding answers and comments. For this, usually every entry is embedded in table data tags. Hence, starting from the digital signature, the table data environment of the same hierarchical level can be extracted and analysed due to the entries considered in the signature. This list of entries is directly connected to the list of variables introduced in the signing process above.

With respect to the signing task, an invisible embedding of digital signatures in practice turned out to be more difficult than expected. The client can only communicate over the interface offered by Moodle, i.e., sites for sending and storing messages. But every message sent to Moodle gets controlled and if necessary is modified. Moodle does not allow to send HTML comments or to apply CSS instructions to some text in order to make it invisible. This, of course, is sensible from the point of view of Moodle programmers, since the possibility of injecting arbitrary code, e.g., in CSS definitions, means an enormous security problem. However, a possible solution to hide the signature is to add it to the `id` attribute of an empty HTML `div` tag. Such tags are only used for structuring web pages and will not be displayed by default. Hence, nothing can be seen by users, but the signature is still there – hidden in the received HTML text.

6.6.5 Conclusion of the Security Agent Project

Within this project, the security agent was meant to be implemented as modular and flexible as possible. Thus, already in this exemplary functional realisation for applying digital signatures to learning contents and communication messages, extensions for integrating smart card support instead of software-driven signing processes are considered. With respect to capabilities of proxy servers as applications on ISO/OSI layer 7, even more security-related tasks are conceivable in practice. For example, these include challenge-response authentication, single sign-on (for clients) even though the infrastructure itself might not regard this comfort aspect, or security checks and filters like stripping malicious XSS code in web pages.

Consequently, the idea of a locally deployed proxy server as personal security secretary provides many useful applications to be implemented on top of the currently given framework. This improves security for learners without increasing the effort of managing all those functions. An automated processing according to programmed rules takes place. Hence, learners will not be distracted from their learning process, since this "secretary" overtakes all time-consuming and recurring interactions.

6.7 Conclusion

This chapter introduced and discussed strategies for dealing with threats for e-learning systems. Recommendations for mitigating discovered threats are given in three layers distinguishing technical mechanisms, organisational deliberations, and user education and support. It can be concluded that organisational security and human behaviour are of major concern for e-learning security, since technical aspects can more easily be established and controlled by protective systems. Consequently, the focus for practical deployments of e-learning systems has to be on appropriate security concepts including users of the system.

In the course of this chapter, several references were applied to draw relations to the fault tree as depicted in Appendix A. This already indicates the value of stated recommendations to cover threats as identified in this research project. However, although it would be desirable to provide recommendations and reliable solutions to all threats as presented in this tree, this is very unlikely especially when considering threats related to human behaviour. Although threats related to technical aspects can be solved easily in most cases, threats related to human fallibility or obliviousness cannot be solved reliably. Notably, the latter case is of relevance for e-learning, since human learners are central for such learning systems. Recommendations concerning education for all users and peer reviews for course creators are reasonable, and therefore, they have to be taken seriously, e.g., to avoid unreliable course contents because of numerous mistakes or inattentive publication of sensitive data by users themselves.

Concluding this chapter, presented recommendations help increase the security level of e-learning systems significantly. With applying these recommendations, all discovered threats will be solved entirely or at least can considerably be mitigated. Bear in mind that different types of e-learning systems as sketched in Section 3.2 implement different functionality. Hence, only a subset of mentioned threats will exist in each type, and as such, only a subset of recommendations is sensibly applicable. Nevertheless, the deployment of a public key infrastructure with the integration of the presented security agent can be reasonable in almost all presented types of e-learning systems. Since the agent can be flexibly extended by further processing modules, almost all security measures that usually require interaction can be established while being kept almost invisible to the users. Thus, distraction for learners by security measures can significantly be reduced while also improving security.

Chapter 7

Conclusion and Future Work

This research project was aimed at a thorough analysis of the information security situation for web-based e-learning systems including the identification of possible threats. The state of research as presented in Chapter 1 showed that security deliberations for e-learning systems were only marginally examined up to this moment. General security investigation was shown to not be sufficient for the special domain of e-learning. Technology-driven approaches can significantly constrain the learning process, such that the primary aim of e-learning systems would be harmed as well. Basing on this cognisance and the lack of scientifically verifiable systematic analyses for e-learning security, open research tasks were formulated as follows:

1. identification of e-learning criteria (with focus on educational requirements),
2. threat analysis and demonstration of case studies, and
3. development of recommendations.

Results in the following will be connected to these objectives and be discussed according to stated evaluation criteria as given in Section 1.4.2.

Objective 1: identification of e-learning criteria

To meet the interdisciplinarity of e-learning, this domain was examined from perspectives of educational science as well as informatics. The conducted analysis has explicitly been started from an e-learning perspective to prevent a technology-driven research possibly neglecting domain specialities. For coping with the fact that no well-accepted definition is available for e-learning systems, activities and tasks for e-learning from perspectives of mentioned disciplines were considered to get an impression of what is commonly expected by an e-learning system. While for the learning component findings from educational science, including educational psychology, were investigated, the “e” implied an examination of informatics subdomains like software engineering and information security for ensuring corresponding system components. The latter perspective primarily resulted in activities to manage system components and functional modules, also taking care of distributed architectures. Thus, functionality, common activities, and related requirements were extracted from related literature, including collaborative systems, for elaborating a definition of e-learning systems to be used in this thesis. Concerning e-learning literature, a categorisation related to five design criteria was stated for classifying activities. These criteria primarily represent expectations and demands by learners, as well as core components of e-learning systems to meet learning theories (cf. Chap. 2). With respect to the amount of available research for e-learning and informatics subdomains, the activity-based definition outlines a complex system for e-learning with high probability of being vulnerable. Mutual influences and dependencies, furthermore, were introduced by learning activities and related demands. Corresponding restraints for security means and for the general construction of e-learning systems were stated in favour of user acceptance and support in the learning process. These limitations foremost are related to privacy demands and to distraction from learning. For preparing the actual threat analysis, assets for e-learning were identified on the basis of introduced learning activities and tasks (cf. Chap. 3). This investigation of relevant assets included the distinction of e-learning systems in five different types, where each of these types is related to different sets of functionality and corresponding demands for roles to be managed.

Objective 2: threat analysis and demonstration of case studies

For the actual threat analysis using a systematic approach to achieve reasonable results, the standardised fault tree analysis (FTA) approach was shown to be suitable. Among a set of most commonly used methodologies this approach turned out to be the best-fitting one for analysing security in web-based e-learning systems (cf. Chap. 4). Applying this systematic and recursive process, events related to actual threats for e-learning were collected and brought into relation with each other for presenting the final fault tree (cf. App. A). In opposition to related works (cf. Chap. 2), the threat analysis of this thesis focused on a systematic procedure with verifiable results. Hence, available evaluation catalogues and databases of common weaknesses and attack patterns were analysed for uncovering threats. These listings built the base for the investigation, such that e-learning specific threats could be extracted and relations be drawn among these threats, including mutual dependencies. This approach has proved to be successful with respect to rare and rather unusual attack patterns. The fact of using well-maintained databases of such patterns as being managed and extended perpetually by large expert communities drastically reduced the probability of neglecting serious threats. Besides this, basing the analysis on assets as deduced from an organisational perspective on e-learning has shown to be advantageous, since discovered threats, which are specific to these assets, can easily be transferred to practically used systems. This was verified by case studies. Theoretical analysis results were evaluated and refined by applying them to the open-source learning management systems Moodle and Ilias (cf. Chap. 5). These systems were analysed and compared in a general way, and specific conceptual problems were discussed with connections to formerly uncovered theoretical threats. During the process of this research project, there were significant changes observable in both selected learning management systems as objects of comparison. Security in this period of time gained in importance in almost all computer scientific areas, such that the notion of security for corresponding developers also changed. To cope with this, the case studies did not only focus on the security concept at a specific point in time, but rather compared these systems concerning their development during the run-time of this project. Besides the practical presentation of weaknesses related to theoretically introduced threats, conceptual shortcomings in these systems were presented to illustrate the necessity for a sound security concept and disadvantages related to an insufficient consideration of user input (cf. [Eib08d, Eib09c]). Weaknesses and security holes as discovered during this analysis, e.g., sensitive file disclosure exploiting the TEX filter in Moodle [Eib09b], were reported to and discussed with corresponding developers to support them in fixing this issue.

Objective 3: development of recommendations

In Chapter 6 former results and requirements for a usable and secure e-learning system were connected to strategies and concrete recommendations for improving security in e-learning systems. As web-based systems they are affected by threats of different kinds, i.e., technical, organisational, and user-related ones. All found threats could be covered by recommendations, but the effect and applicability of the recommendations are very different. While recommendations for technical threats are promising, recommendations targeting human failure must be considered critically. Although their application is sensible, the improvement on the actual security gain is not reliable due to non-deterministic behaviour of human beings. Basing on introduced activities, it was shown that for e-learning security the human factor is of significant relevance. Thus, integration of security education, provision of user manuals, and guidance were considered, as well as suggestions for involving users to increase participation and enable better identification with the deployed system. Furthermore, the reduction of details in error messages was discussed to not provide excessive information about internal data structures. Decreased usability due to fewer details in software dialogues mainly can be compensated by increased support. However, stated recommendations require efforts to be implemented. Also, deployed security measures might require additional interactions with users. Thus, to further improve security without burdening or distracting learners, i.e., meet restraints concerning distraction by security means, a security agent in the form of a locally deployed proxy server was presented to overtake security critical tasks from learners. With this, the majority of security-related interaction can be transferred to this agent, and distraction for learners can be reduced. A prototype of this security agent for digitally signing and verifying e-learning contents and communication data was finished and made available to the public in May 2009.

Basing on research activities and results of this dissertation thesis, some questions have risen and provide links for subsequent research. As discussed in Chapter 5, only open-source learning management systems were examined. Although open-source has specific advantages for analysing security concepts and implementation quality, these systems only represent a small amount of all available systems. Discussions at national and international conferences, where the author has presented conceptual weaknesses of regarded open-source learning management system, resulted in uniform confirmations by colleagues, who work with corresponding commercial systems, that similar weaknesses can also be found there. Nevertheless, analysis results could not be transferred and verified accordingly, such that this is left open for future work. In addition, the security analysis of this dissertation thesis was focusing on the discovery of threats to provide a base for developing and implementing sophisticated and adequate security means for e-learning. Neither probabilities of threat occurrence nor their impact in case of incidents were considered. Thus, discovered threats cannot be quantitatively analysed and compared yet, and as such, no prioritisation for mitigation and no classification concerning the actual relevance in practical e-learning systems is possible. However, other analysis approaches, like the failure modes and effects analysis (FMEA), can be started on top of given results to continue this research – including the consideration of risks for presented threats. Finally, as sketched in Section 2.7, high levels of security most likely require an increased amount of sophisticated security means that additionally demand user interaction, and therefore, can have negative influences on learning due to increased distraction. Presented recommendations, in combination with the security agent, can be assumed to be applicable without increased distraction for learners from their learning progress. But in general, the optimal compromise between security improvements and acceptable distraction has not been empirically investigated, yet. Thus, studies with different amounts of security measures and corresponding security levels will have to be examined and compared concerning their influence on the learning progress and usability of the system.

Bibliography

[Aba03] Abadi, M.: *Private Authentication.* In: R. Dingledine; P. Syverson (eds.), *PET 2002*, Springer, Berlin Heidelberg, 2003, LNCS 2483, pp. 27–40.

[Adv09] Advanced Distributed Learning: *Sharable Content Object Reference Model (SCORM) 2004.* 4th ed., Mar. 2009.

[And01] Anderson, R.J.: *Security Enginieering: A Guide to Building Dependable Distributed Systems.* Wiley Computer Publishing, New York, 2001.

[Anl02] Anley, C.: *Advanced SQL Injection In SQL Server Applications.* Tech. rep., NGSSoftware Insight Security Research (NISR), 2002.

[ÅSS07] Åhlfeldt, R.M.; Spagnoletti, P.; Sindre, G.: *Improving the Information Security Model by using TFI.* In: Venter et al. [VEL$^+$07], pp. 73–85, proceedings of IFIP sec.

[Bal00] Balzert, H.: *Lehrbuch der Software-Technik, Band 1, Software-Entwicklung.* Spektrum Akademischer Verlag, 2000, 2nd ed.

[Bar09] Barisani, A.: *mimeTeX and mathTeX buffer overflows and command injection.* Bug Traq mailing list, Jul. 2009, URL: `http://www.securityfocus.com/archive/1/504919`, [04-10-2009].

[Bau06] Baumgartner, P.: *Unterrichtsmethoden als Handlungsmuster – Vorarbeiten zu einer didaktischen Taxonomie für E-Learning.* In: M. Mühlhäuser; G. Rößling; R. Steinmetz (eds.), *DeLFI 2006: 4. e-Learning Fachtagung Informatik*, Köllen Druck+Verlag, Bonn, 2006, LNI Band 87, ISBN 978-3-88579-181-2, pp. 51–62.

[BBR62] Beal, G.; Bohlen, J.; Raudsbaugh, J.: *Leadership and Dynamic Group Action.* Iowa State University Press, Ames, Iowa, 1962.

[BDF$^+$06] Borcea, K.; Donker, H.; Franz, E.; Pfitzmann, A.; Wahrig, H.: *Towards Privacy-Aware eLearning.* In: G. Danezis; D. Martin (eds.), *PET 2005*, Springer, 2006, no. 3856 in LNCS, pp. 167–178.

[BDRW03] Bode, A.; Desel, J.; Rathmayer, S.; Wessner, M. (eds.): *DeLFI 2003: Die 1. E-Learning Fachtagung Informatik*, LNI, Köllen Druck+Verlag, Bonn, 2003.

[Bel74] Bell, D.: *Secure Computer Systems: A Refinement of the Mathematical Model.* Tech. Rep. ESD-TR-73-278, Vol. III, Electronic Systems Division, Air Force Systems Command, Hanscom AFB, Bedford, Massachusetts, Apr. 1974.

[BG02] Bruns, B.; Gajewski, P.: *Multimediales Lernen im Netz: Leitfaden für Entscheider und Planer.* Springer, Berlin, 2002, 3rd ed., ISBN 3-540-42477-6.

[BHM05] Baumgartner, P.; Häfele, H.; Maier-Häfele, K.: *Evaluation von Lernplattformen: Verfahren, Ergebnisse und Empfehlungen (Version 1.3)*. Tech. rep., Bundesministerium für Bildung, Wissenschaft und Kultur (BMBWK), Österreich, 2005, URL: `http://www.bildung.at/statisch/bmbwk/lernplattformen_evaluation_und_ergebnisse_1_bis_3.pdf`, [06-12-2005].

[BL73] Bell, D.; LaPadula, L.: *Secure Computer Systems: Mathematical Foundations*. Tech. Rep. ESD-TR-73-278, Vol. I, Electronic Systems Division, Air Force Systems Command, Hanscom AFB, Bedford, Massachusetts, Nov. 1973.

[BL76] Bell, D.E.; LaPadula, L.J.: *Secure Computer Systems: Unified Exposition and Multics Interpretation*. Tech. Rep. ESD-TR-75-306, US Air Force, Mar. 1976.

[BM93] Biggs, J.B.; Moore, P.J.: *The Process of Learning*. Prentice Hall, New York, 1993, 3rd ed.

[BP07] Bauer, B.; Patrick, A.: *A Human Factors Extension to the Seven-Layer OSI Reference Model*. In: The IAENG International Journal of Computer Science, Oct. 2007.

[BSI05] BSI: *IT-Grundschutz Catalogues: Version 2005*. Federal Office for Information Security (BSI), Berlin, 2005, URL: `https://www.bsi.bund.de/cae/servlet/contentblob/479604/publicationFile/28021/it-grundschutz-kataloge_2005_pdf_en_zip.zip`, [27-01-2010].

[Bud04] Buddensiek, D.: *Barrierefreiheit im Internet*. In: Engels Seehusen [ES04], pp. 27–30.

[CAL05] Caeiro-Rodríguez, M.; Anido-Rifón, L.; Llamas-Nistal, M.: *Improving the modelling of heterogeneous learning activities*. In: Samways [Sam05], proceedings on CD-ROM, 387.pdf.

[Cen02] Coordination Center, C.: *Securing Networks Systematically – the SKiP Method*. Tech. rep., Carnegie Mellon University, 2002, URL: `http://www.cert.org/archive/pdf/SKiP.pdf`, [02-05-2008].

[Cha81] Chaum, D.: *Untraceable Electronic Mail, Return Address, and Digital Pseudonyms*. In: Communications of the ACM, 24(2), 1981, pp. 84–88.

[Cou77] Courtney, R.: *Security risk assessment in electronic data processing*. In: *AFIPS Conference Proceedings of the National Computer Conference 46*, AFIPS, 1977, pp. 97–104.

[Cre05] Cress, U.: *Why Member Portraits Can Undermine Participation*. In: Koschmann et al. [KSC05], pp. 86–90.

[CS05a] Cárdenas, R.G.; Sánchez, E.M.: *Security Challenges of Distributed e-Learning Systems*. In: F. Ramos; et al. (eds.), *Fifth IEEE Int. Symp. and School on Advanced Distributed Systems (ISSADS) 2005*, Springer, Berlin, 2005, no. 3563 in LNCS, pp. 538–544.

[CS05b] Coppeto, T.; Sim, S.: *Service Oriented Architectures with O.K.I.* JA-SIG Conference, presentation, Dec. 2005, URL: `http://web.princeton.edu/sites/isapps/jasig/2005WinterAustin/presentations/SOACopetto.ppt`, [05-02-2010].

[Dah06] Dahm, M.: *Grundlagen der Mensch-Computer-Interaktion*. Pearson Verlag, München, 2006.

[Dew38] Dewey, J.: *Experience and Education*. Mcmillan, New York, 1938.

[Dhi07] Dhillon, G.: *Principles of Information Systems Security: Text and Cases*. John Wiley & Sons, Inc., 2007, ISBN 978-0-471-45056-6.

[DIN81] DIN 25424: *Fehlerbaumanalyse.* Deutsches Institut für Normung, 1981, teil 1: 1981; Teil 2: 1990.

[DIN06] DIN EN 60812: *Analysetechniken für die Funktionsfähigkeit von Systemen – Verfahren für die Fehlzustandsart- und -auswirkungsanalyse (FMEA).* Deutsches Institut für Normung, IEC 60812:2006, 2006.

[Dit00] Dittmann, J.: *Digitale Wasserzeichen.* Springer, Berlin, 2000, ISBN 3-540-66661-3.

[Eck03] Eckert, C.: *Sicherheit und E-Learning.* In: R. Grimm (ed.), *Workshop "E-Learning: Beherrschbarkeit und Sicherheit"*, 2003, 9, ISSN 1617-9048, pp. 43–50.

[Eck04] Eckert, C.: *IT-Sicherheit.* Oldenbourg Wissenschaftsverlag, München, 2004, 3rd ed., ISBN 3-486-20000-3.

[Edu02] Eduworks Corporation: *What is the Open Knowledge Initiative?* white paper, Sep. 2002, URL: `http://web.mit.edu/oki/learn/whtpapers/OKI_white_paper_120902.pdf`, [07-03-2006].

[Eib07] Eibl, C.J.: *Information Security in E-Learning.* In: C. Abbott; Z. Lustigova (eds.), *Information Technologies for Education and Training*, University of Prague, 2007, pp. 204–213.

[Eib08a] Eibl, C.J.: *Entwicklung von E-Learning-Designkriterien und Implikationen für die Informationssicherheit.* In: Seehusen et al. [SLF08], pp. 377–388.

[Eib08b] Eibl, C.J.: *Informationssicherheit in E-Learning-Systemen.* In: S. Schubert; P. Stechert (eds.), *Bildungskonzepte für Internetworking und eingebettete Mikrosysteme*, universi Verlag, Siegen, 2008, ISBN 978-3-936533-28-6, pp. 141–165.

[Eib08c] Eibl, C.J.: *Risk Analysis towards Secure E-Learning.* In: S. Wheeler; A. Kassam; D. Brown (eds.), *LYICT 2008*, Proc. on CD-ROM, 2008, ISBN 978-3-901882-29-6.

[Eib08d] Eibl, C.J.: *Vertraulichkeit persönlicher Daten in Lern-Management-Systemen.* In: Seehusen et al. [SLF08], pp. 317–328.

[Eib09a] Eibl, C.J.: *Ereignisbasierte und konzeptuelle Schwachstellen in E-Learning-Systemen.* In: Schwill Apostolopoulos [SA09], pp. 91–102.

[Eib09b] Eibl, C.J.: *Moodle: Sensitive File Disclosure.* Bug Traq mailing list, Mar. 2009, URL: `http://www.securityfocus.com/archive/1/502231`, [04-10-2009].

[Eib09c] Eibl, C.J.: *Privacy and Confidentiality in E-Learning Systems.* In: M. Perry; H. Sasaki; M. Ehmann; G. Ortiz; O. Dini (eds.), *Fourth International Conference on Internet and Web Applications and Services (ICIW 2009)*, IEEE Computer Society Press, 2009, ISBN 978-0-7695-3613-2.

[Eib09d] Eibl, C.J.: *Sicherheitsprobleme dynamischer Erweiterbarkeit in E-Learning-Systemen.* In: Schwill Apostolopoulos [SA09], pp. 103–113.

[EKXY03] El-Khatib, K.; Korba, L.; Xu, Y.; Yee, G.: *Privacy and Security in E-Learning.* In: Int. Journal of Distance Education, 1(4), Oct. 2003, pp. 1–19, URL: `http://www.igi-online.com/downloads/pdf/itj2493_cfvs4lcc1e.pdf`, [14-03-2006].

[EMSW07] Eibl, C.; Magenheim, J.; Schubert, S.; Wessner, M. (eds.): *DeLFI 2007: Die 5. e-Learning Fachtagung Informatik*, LNI Band 111, Köllen Druck+Verlag, Bonn, 2007.

[Eri03] Erickson, J.: *Hacking: The Art of Exploitation.* No Starch Press, 2003.

[ES04] Engels, G.; Seehusen, S. (eds.): *DeLFI 2004: Die 2. E-Learning Fachtagung Informatik*, LNI, Köllen Druck+Verlag, Bonn, 2004.

[ES08a] Eibl, C.J.; Schubert, S.: *Development of E-Learning Design Criteria with Secure Realization Concepts.* In: R. Mittermeir; M. Sysło (eds.), *ISSEP 2008*, Springer, Berlin, 2008, *LNCS*, vol. 5090, pp. 327–336.

[ES08b] Eibl, C.J.; Schubert, S.: *Development of E-Learning Design Criteria with Secure Realization Concepts.* In: M. Kendall; B. Samways (eds.), *Learning to Live in the Knowledge Society*, Springer, Boston, 2008, *IFIP*, vol. 281, pp. 361–362.

[ESS06] Eibl, C.J.; von Solms, S.H.; Schubert, S.: *A Framework for Evaluating the Information Security of E-Learning Systems.* In: V. Dagienė; R. Mittermeir (eds.), *Information Technologies at School*, 2006, pp. 83–94, proceedings of the Second International Conference "Informatics in Secondary Schools: Evolution and Perspectives".

[ESS07] Eibl, C.J.; von Solms, S.H.; Schubert, S.: *Development and Application of a Proxy Server for Transparently, Digitally Signing E-Learning Content.* In: Venter et al. [VEL$^+$07], pp. 181–192, proceedings of IFIP sec.

[FGM$^+$99] Fielding, R.; Gettys, J.; Mogul, J.; Frystyk, H.; Masinter, L.; Leach, P.; Berners-Lee, T.: *Hypertext Transfer Protocol – HTTP/1.1.* Request for Comments (RFC) 2616, 1999.

[FU04] Ferstl, O.; Ulrich, C.: *Eine MultiView-Benutzungsoberfläche für integrierte Lernumgebungen.* In: Engels Seehusen [ES04], pp. 103–114.

[GB01] Geer, R.; Barnes, A.: *Teaching and Learning through Technology-mediated Interaction.* In: Watson Anderson [WA01], pp. 247–256.

[GCC$^+$05] Gee, Q.; Carr, L.; Conole, G.; Grange, S.; Hall, W.; Wills, G.: *Building a virtual university for orthopaedics.* In: Samways [Sam05], proc. on CD-ROM, 333.pdf.

[GI00] GI: *Empfehlungen der Gesellschaft für Informatik e. V. für ein Gesamtkonzept zur informatischen Bildung an allgemein bildenden Schulen.* In: Informatik Spektrum, 23(6), Dec. 2000.

[Gra03] Graf, F.: *Lernspezifische Sicherheitsmechanismen in Lernumgebungen mit modularem Lernmaterial.* Ph.D. thesis, TU Darmstadt, 2003.

[Gro03] Grobler, C.: *A Model to assess the Information Security status of an organization with special reference to the Policy Dimension.* Master's thesis, Rand Afrikaans University, Oct. 2003.

[GRS99] Goldschlag, D.; Reed, M.; Syverson, P.: *Onion Routing for Anonymous and Private Internet Connections.* In: Communications of the ACM, 42(2), 1999, pp. 39–41.

[GRV05] Gürgens, S.; Rudolph, C.; Voigt, H.: *On the security of fair non-repudiation protocols.* In: International Journal of Information Security, 4(4), Oct. 2005, pp. 253–262, ISSN 1615-5262.

[Gud01] Gudjons, H.: *Pädagogisches Grundwissen.* Klinkhardt, Regensburg, 2001, 7th ed.

[Gui09] Guimarães, B.D.A.: *Advanced SQL injection to operating system full control.* Black Hat 2009, White Paper, 2009, URL: `http://www.blackhat.com/presentations/bh-europe-09/Guimaraes/BlackHat-Europe-09-Damele-A-G-Advanced-SQL-injection-whitepaper.pdf`, [05-02-2010].

[Ham02] Hamid, A.A.: *e-Learning: Is it the "e" or the learning that matters?* In: Internet and Higher Education, 4(3), 2002, pp. 311–316.

[HB03] Hinze, U.; Blakowski, G.: *Soziale Eingebundenheit als Schlüsselfaktor im E-Learning.* In: Bode et al. [BDRW03], pp. 57–66.

[HBH+04] Haake, A.; Bourimi, M.; Haake, J.M.; Schümmer, T.; Landgraf, B.: *Endbenutzergesteuerte Gruppenbildung in gemeinsamen Lernräumen.* In: Engels Seehusen [ES04], pp. 235–246.

[Her06] Herzog, P.: *OSSTMM 2.2. - The Open Source Security Testing Methodology Manual.* Tech. rep., Institute for Security and Open Methodologies (ISECOM), Dec. 2006.

[HLT05] Haake, J.M.; Lucke, U.; Tavangarian, D. (eds.): *DeLFI 2004: Die 3. E-Learning Fachtagung Informatik*, LNI, Köllen Druck+Verlag, Bonn, 2005.

[HQW08] Hoffmann, A.; Quast, A.; Wismüller, R.: *Online-Übungssystem für die Programmierausbildung zur Einführung in die Informatik.* In: Seehusen et al. [SLF08], pp. 173–184.

[HS07] Herzog, A.; Shahmehri, N.: *Usability and Security of Personal Firewalls.* In: Venter et al. [VEL+07], pp. 37–48, proceedings of IFIP sec.

[HSP03] Hentea, M.; Shea, M.J.; Pennington, L.: *A Perspective on Fulfilling the Expectations of Distance Education.* In: CITC4'03, (4), Oct. 2003, pp. 160–167.

[HT83] Hertz, D.B.; Thomas, H.: *Risk analysis and its applications.* Wiley Computer Publishing, Chichester, 1983, ISBN 0-471-10145-1.

[HW05] Hammond, M.; Wiriyapinit, M.: *Using online discussion to support teaching and learning: opportunities and challenges.* In: Samways [Sam05], proceedings on CD-ROM, 208.pdf.

[HW08] Hoffmann, A.; Wismüller, R.: *Sicherheitskonzept für elektronische Prüfungen an Hochschulen auf Basis eines ticketbasierten, virtuellen Dateisystems.* In: Seehusen et al. [SLF08], pp. 197–208.

[HWB09] Hoffmann, A.; Wismüller, R.; Bode, M.: *Realisierung eines Sicherheits- und Rechtemanagements für elektronische Prüfungen an Hochschulen mittels Software-Proxy.* In: Schwill Apostolopoulos [SA09], pp. 271–282.

[IEC01] IEC 60617-12: *Graphical symbols for diagrams.* IEC, 2001.

[IEE01] IEEE Computer Society: *IEEE P1484.1/D9, 2001-11-30 Draft Standard for Learning Technology – Learning Technology Systems Architecture (LTSA).* IEEE Computer Society, 2001, URL: `http://ltsc.ieee.org/wg1/files/IEEE_1484_01_D09_LTSA.pdf`, [05-02-2010].

[IMS01] IMS Global Learning Consortium, Inc.: *Learner Information Package.* Instructional Management System, 2001, URL: `http://www.imsglobal.org/profiles/index.html`, [08-01-2010].

[IMS06a] IMS Global Learning Consortium, Inc.: *Content Packaging Specification.* Instructional Management System, 2006, URL: `http://www.imsglobal.org/content/packaging/index.html`, [23-02-2006].

[IMS06b] IMS Global Learning Consortium, Inc.: *Question & Test Interoperability Specification.* Instructional Management System, 2006, URL: `http://www.imsglobal.org/question/index.html`, [23-02-2006].

[ISO89] ISO 7498-2: *Information processing systems – Open systems interconnection – Basic reference model – Part 2: Security Architecture.* International Organization for Standardization (ISO), 1989.

[ISO94] ISO 7498-1: *Information processing systems – Open systems interconnection – Basic reference model: The Basic Model.* International Organization for Standardization (ISO), 1994, 2nd ed.

[ISO01] ISO/IEC 9126-1: *Software engineering – Product quality – Part 1: Quality model.* International Organization for Standardization (ISO), 2001.

[ISO05] ISO/IEC 15408: *Common Criteria for Information Technology Security Evaluation.* Parts I-III, Version 2.3, Aug. 2005.

[ITS91] *Information Technology Security Evaluation Criteria (ITSEC)*, 1991, harmonised Criteria of France, Germany, the Netherlands, and the United Kingdom.

[ITU91] ITU CCITT: *Security Architectures for Open Systems Interconnection for CCITT Applications – Recommendation X.800.* International Telecommunication Union (ITU), The international telegraph and telephone consultative committee (CCITT), Geneva, 1991.

[Jev06] Jevsikova, T.: *Localization and Internationalization of Web-Based Learning Environment.* In: R. Mittermeir (ed.), *ISSEP 2006*, Springer-Verlag, 2006, *LNCS*, vol. 4226, pp. 310–318.

[Kaj03] Kajava, J.: *Security in e-Learning: the Whys and Wherefores.* In: *European Intensive Programme on Information and Communication Technologies Security (IPICS'2003), 4th Winter School*, 2003.

[Kam08] Kaminsky, D.: *Black Ops 2008: It's The End Of The Cache As We Know It.* Presentation at Black Hat Conference 2008, Aug. 2008, URL: `http://www.doxpara.com/DMK_BO2K8.ppt`, [12-03-2009].

[KBC05] Kohno, T.; Broida, A.; Claffy, K.: *Remote Physical Device Fingerprinting.* In: IEEE Transactions on Dependable and Secure Computing, 2(2), Apr. 2005, pp. 93–108.

[KBS05] Krafzig, D.; Banke, K.; Slama, D.: *Enterprise SOA – Service-Oriented Architecture Best Practices.* Pearson Education, 2005, ISBN 0-13-146575-9.

[Ker01] Kerres, M.: *Multimediale und telemediale Lernumgebungen.* Oldenbourg Wissenschaftsverlag, 2001, 2nd ed.

[KHS05] Knowles, M.; Holton, E.; Swanson, R.: *The Adult Learner – the definitive classic in adult education and human resource development.* Elsevier, Amsterdam, 2005, 6th ed.

[Kle06] Klein, A.: *HTTP Message Splitting, Smuggling and Other Animals.* Presentation on OWASP AppSec Europe, May 2006, URL: `http://www.owasp.org/images/1/1a/OWASPAppSecEU2006_HTTPMessageSplittingSmugglingEtc.ppt`, [05-02-2010].

[KR04] Kienle, A.; Ritterskamp, C.: *Rollenbasierte Kooperationsunterstützung in CSCL-Umgebungen.* In: Engels Seehusen [ES04], pp. 223–234.

[KSC05] Koschmann, T.; Suthers, D.D.; Chan, T.W. (eds.): *CSCL 2005 – The Next 10 Years!*, Proceedings of CSCL 2005, Lawrence Erlbaum Associates, Inc., Mahwah, New Jersey, USA, 2005.

[KV02a] Kajava, J.; Varonen, R.: *Internet Security and E-Teaching.* In: Riedling [Rie02], pp. 57–66.

[KV02b] Kajava, J.; Varonen, R.: *Towards a Transparent University: The Role of Cryptography, Control Measures and the Human Users.* In: Riedling [Rie02], pp. 67–76.

[LB73] LaPadula, L.; Bell, D.: *Secure Computer Systems: A Mathematical Model.* Tech. Rep. ESD-TR-73-278, Vol. II, Electronic Systems Division, Air Force Systems Command, Hanscom AFB, Bedford, Massachusetts, Nov. 1973.

[Lin77] Linskie, R.: *The Learning Process: Theory and Practice.* Litton Educational Publishing, New York, 1977.

[Lip02] Lipponen, L.: *Exploring Foundations for Computer-Supported Collaborative Learning.* In: Stahl [Sta02], pp. 72–81.

[LTZ$^+$08] Lehsten, P.; Thiele, A.; Zilz, R.; Dressler, E.; Zender, R.; Lucke, U.; Tavangarian, D.: *Dienste-basierte Kopplung von virtueller und Präsenzlehre.* In: Seehusen et al. [SLF08], pp. 77–88.

[LW91] Lave, J.; Wenger, E.: *Situated learning: Legitimate peripheral participation.* Cambridge University Press, New York, 1991.

[MBPC09] Martin, B.; Brown, M.; Paller, A.; Christley, S.: *2009 CWE/SANS Top 25 most Dangerous Programming Errors.* online: http://cwe.mitre.org/top25, Jan. 2009.

[MC01] McCarthy, M.P.; Campbell, S.: *Security transformation – digital Defense Strategies to Protect Your Company's Reputation and Market Share.* McGraw-Hill, New York, 2001.

[MD05] Munoz, K.; Duzer, J.: *Blackboard vs. Moodle – A Comparison of Satisfaction with Online Teaching and Learning Tools*, Feb. 2005, URL: `http://www.humboldt.edu/~jdv1/moodle/all.htm`, [07-03-2006].

[Mey88] Meyer, H.: *Unterrichtsmethoden – II: Praxisband.* Cornelsen Scriptor, Berlin, 1988, 2nd ed., ISBN 978-3-589-20851-7.

[MH05] Maier-Häfele, K.; Häfele, H.: *Open-Source-Werkzeuge für e-Trainings.* managerSeminare Verlags GmbH, Bonn, 2005.

[MIT10] MITRE Corporation: *Common Attack Pattern Enumeration and Classification (CAPEC).* Online catalogue, 2010, URL: `http://capec.mitre.org/data/graphs/1000.html`, [27-01-2010].

[MK07] Meucci, M.; Keary, E. (eds.): *OWASP Testing Guide 2.0*, Open Web Application Security Project (OWASP Foundation), 2007.

[MS03] Mitnick, K.; Simon, W.: *Die Kunst der Täuschung.* mitp-Verlag, Bonn, 2003.

[Mul03] Mullen, T.M.: *Lost in Translation.* Essay, SecurityFocus, online, Sep. 2003, URL: `http://www.securityfocus.com/columnists/186`, [30-11-2009].

[Mur06] Murdoch, S.J.: *Hot or Not: Revealing Hidden Services by their Clock Skew.* In: *Proceedings of CCS'06*, ACM, 2006.

[Nie94] Nielsen, J.: *Usability Engineering.* Morgan Kaufman Publishers, San Francisco, 1994.

[OEC01] OECD: *E-Learning – The Partnership Challenge.* Centre for educational research and innovation, OECD Publications, Paris, France, 2001, ISBN 92-64-18693-X.

[One96] One, A.: *Smashing The Stack For Fun And Profit.* Phrack online magazine, Volume Seven, Issue Forty-Nine, File 14 of 16, Nov. 1996, URL: `http://insecure.org/stf/smashstack.html`, [06-12-2005].

[Par81] Parker, D.B.: *Computer Security Management.* Reston Publishing Company, Inc., Reston, Virginia, 1981, ISBN 0-8359-0905-0.

[PK08] Pilosov, A.; Kapela, T.: *Stealing The Internet.* Slides for Defcon 16, Las Vegas, NV, Aug. 2008, URL: `https://www.defcon.org/images/defcon-16/dc16-presentations/defcon-16-pilosov-kapela.pdf`, [05-02-2010].

[POP08] Parata, A.; Ongaro, F.; Pellerano, G.: *Moodle 1.9.3 Remote Code Execution.* Security Focus – BugTraq, Dec. 2008, URL: `http://www.securityfocus.com/archive/1/499172/30/840/threaded`, [05-02-2010].

[Rie02] Riedling, E. (ed.): *VIEWDET' 2002. Vienna International Working Conference – eLearning and eCulture*, Band 162, Oesterreichische Computer Gesellschaft (OCG), 2002, ISBN 3-85403-162-9.

[Ris05] Ristic, I.: *Apache Security.* O'Reilly Media, 2005.

[Rot63] Roth, H.: *Pädagogische Psychologie des Lehrens und Lernens.* Hermann Schroedel Verlag KG, Hannover, 1963, 14th ed.

[RS75] Rockart, J.R.; Scott Morton, M.S.: *Computers and the Learning Process in Higher Education.* Carnegie Commission on Higher Education, McGraw-Hill Book Company, 1975.

[RSG97] Reed, M.; Syverson, P.; Goldschlag, D.: *Anonymous Connections and Onion Routing.* In: *Proceeding of the 1997 IEEE Computer Society Symposium on Research in Security and Privacy*, IEEE Computer Society Press, Los Alamitos, 1997, pp. 44–54.

[SA09] Schwill, A.; Apostolopoulos, N. (eds.): *DeLFI 2009: Lernen im Digitalen Zeitalter*, LNI Band 153, Köllen Druck+Verlag, Bonn, 2009.

[Sam05] Samways, B. (ed.): *35 Years of Computers in Education: What Works?*, 8th World Conference on Computers in Education – WCCE '05, IFIP, Proc. on CD-ROM, 2005.

[Sch96] Schneier, B.: *Applied Cryptography.* Wiley Computer Publishing, New York, 1996, 2nd ed.

[Sch99] Schneier, B.: *Attack Trees – Modeling Security Threats.* In: Dr. Dobb's Journal, Dec. 1999, URL: `http://www.schneier.com/paper-attacktrees-ddj-ft.html`, [05-02-2010].

[Sch02] Schulmeister, R.: *Taxonomy of Interactivity in Multimedia – A Contribution to the Actual Metadata Discussion.* In: it+ti – Informationstechnik und Technische Informatik, Oldenbourg Verlag, 44(4), 2002, pp. 193–199.

[Sch03a] Schneier, B.: *Beyond Fear: Thinking Sensibly about Security in an Uncertain World.* Springer-Verlag, New York, 2003.

[Sch03b] Schulmeister, R.: *Lernplattformen für das virtuelle Lernen.* Oldenbourg, München, 2003.

[Sch05] Schulmeister, R.: *Interaktivität in Multimedia-Anwendungen*, Nov. 2005, URL: `http://www.e-teaching.org/didaktik/gestaltung/interaktiv/InteraktivitaetSchulmeister.pdf`, [28.01.2008].

[Sei06] Seibold, H.: *IT-Risikomanagement.* Oldenbourg Wissenschaftsverlag, München, 2006, ISBN 978-3-486-58009-9.

[SES07] Schwidrowski, K.; Eibl, C.J.; Schubert, S.: *Projekt Internetworking und E-Learning.* In: S. Schubert (ed.), *Didaktik der Informatik in Theorie und Praxis (INFOS 2007)*, Köllen Druck+Verlag, Bonn, 2007, LNI Band 112, ISBN 978-3-88579-206-2, pp. 331–332.

[SES08] Schwidrowski, K.; Eibl, C.J.; Schubert, S.: *Internetworking und E-Learning – Bildungsanforderungen und Interaktionsstufen.* In: Navigationen: Interaktionen, 8(1), 2008, pp. 141–158.

[SES09] Schwidrowski, K.; Eibl, C.J.; Schubert, S.: *Konzept des E-Learning-Kurses "Internetworking".* In: *Grundlagen Multimedialen Lehrens und Lernens (GML^2) 2007*, Universitätsverlag der TU Berlin, 2009, pp. 167–172.

[SG06] Stacey, E.; Gerbic, P.: *Teaching for blended learning. How is ICT impacting on distance and on campus education?* In: D. Kumar; J. Turner (eds.), *Education for the 21st Century-Impact of ICT and Digital Resources*, Springer, Boston, 2006, Proceedings of the IFIP 19th World Computer Congress, TC-3, Education, pp. 225–234.

[SGF02] Stoneburner, G.; Goguen, A.; Feringa, A.: *Risk Management Guide for Information Technology Systems.* Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology (NIST), Special Publication 800-30, Jul. 2002.

[SH06] Schreiber, T.; Hoffmann, A.: *Sicherheit von Webanwendungen – Maßnahmenkatalog und Best Practices.* Tech. rep., Bundesamtes für Sicherheit in der Informationstechnik, Bonn, Aug. 2006.

[Shn92] Shneiderman, B.: *Designing the User Interface.* Addison-Wesley, Reading, Massachusetts, 1992, 2nd ed.

[SLF08] Seehusen, S.; Lucke, U.; Fischer, S. (eds.): *DeLFI 2008: Die 6. e-Learning Fachtagung Informatik*, LNI Band 132, Köllen Druck+Verlag, Bonn, 2008.

[SLM06] Sánchez, D.D.; Lopez, A.M.; Mendoza, F.A.: *A Smart Card Solution for Access Control and Trust Management for Nomadic Users.* In: J. Domingo-Ferrer; J. Posegga; D. Schreckling (eds.), *CARDIS 2006*, Springer, 2006, no. 3928 in LNCS, pp. 62–77.

[Sol05] von Solms, S.H.: *Information Security Governance in ICT Based Educational Systems.* Fourth International Conference on ICT and Higher Education, Siam University, Bangkok, Thailand, Sep. 2005, URL: `http://www.die.informatik.uni-siegen.de/inel/english/BangkokSept05FIN.pdf`, [14-03-2006].

[Spr06] van Sprundel, I.: *Unusual bugs.* Präsentation auf RuxCon Konferenz, 2006, URL: `http://www.ruxcon.org.au/files/2006/unusual_bugs.pdf`, [13-03-2007].

[SS99] Shamir, A.; van Someren, N.: *Playing 'Hide and Seek' with Stored Keys.* In: M. Franklin (ed.), *FC'99*, Springer-Verlag Berlin Heidelberg, 1999, *LNCS*, vol. 1648, pp. 118–124.

[SSK03] Schmid, M.; Stynes, J.; Kröger, R.: *Architektur einer dienstebasierten, personalisierbaren Laufzeitumgebung für Lernumgebungen.* In: Bode et al. [BDRW03], pp. 225–229.

[SSSE07] Schäfer, J.; Schubert, S.; Schwidrowski, K.; Eibl, C.J.: *Digitale Literatur und Kunst – Blended Learning zu ästhetischen Prozessen in und mit Informatiksystemen.* In: E-Learning und Literatur. Massenmedien und Kommunikation (MuK), (166/167), 2007, pp. 61–78, ISSN 0271-3271.

[Sta01] Stacey, E.: *Social Presence Online: Networking Learners at a Distance.* In: Watson Anderson [WA01], pp. 39–48, proc. of WCCE 2001.

[Sta02] Stahl, G. (ed.): *Computer Support for Collaborative Learning: Foundations for a CSCL Community*, Proceedings of CSCL 2002, Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, USA, 2002.

[Stö78] Stöcker, K.: *Neuzeitliche Unterrichtsgestaltung.* Ehrenwirth Verlag, München, 1978, 17th ed.

[Swa01] Swanson, M.: *Security Self-Assessment Guide for Information Technology Systems.* Tech. Rep. Special Publication 800-26, National Institute of Standards and Technology (NIST), Nov. 2001.

[Tra01] Traxler, J. (ed.): *Hack Proofing Your Web Applications.* Syngress Publishing, Rockland, MA, 2001, ISBN 1-928994-31-8.

[VEL+07] Venter, H.; Eloff, M.; Labuschagne, L.; Eloff, J.; von Solms, R. (eds.): *New Approaches for Security, Privacy and Trust in Complex Environments*, IFIP TC-11, Springer Science+Business Media, Boston, 2007, proceedings of IFIP sec.

[VGRH81] Vesely, W.; Goldberg, F.; Roberts, N.; Haasl, D.: *Fault Tree Handbook.* US Nuclear Regulatory Commission, NUREG-0492, 1981.

[VK83] Voydock, V.; Kent, S.: *Security Mechanisms in High-Level Network Protocols.* In: ACM Computing Surveys, 15(2), Jun. 1983, pp. 135–171, URL: `http://portal.acm.org/ft_gateway.cfm%3Fid%3D356913%26type%3Dpdf%26dl%3Dportal%26dl%3DACM%26CFID%3D11111111%26CFTOKEN%3D2222222`, [14-03-2006].

[Vyg78] Vygotsky, L.: *Mind in society – The Development of Higher Psychological Processes.* Harvard University Press, Cambridge, MA, 1978.

[WA01] Watson, D.; Anderson, J. (eds.): *Networking the Learner: Computers in Education*, WCCE '01, IFIP, Kluwer Academic Publishers, Boston, 2001, ISBN 1-4020-7133-7.

[Wei05] Weippl, E.R.: *Security in E-Learning.* Springer Science+Business Media, New York, 2005.

[Wes05] Wessner, M.: *Kontextuelle Kooperation in virtuellen Lernumgebungen.* Dissertation TU Darmstadt 2004, Eul Verlag, Cologne, 2005, ISBN 3-89936-416-3.

[Wes08] West, R.: *The Psychology of Security.* In: Communications of the ACM, 51(4), Apr. 2008, pp. 34–40.

[WFS02] Wu, A.S.; Farrell, R.; Singley, M.K.: *Scaffolding Group Learning in a Collaborative Networked Environment.* In: Stahl [Sta02], pp. 245–254.

[WH03] Warren, M.; Hutchinson, W.: *Information Security – An E-learning Problem.* In: W. Zhou; et al. (eds.), *ICWL 2003*, Springer, 2003, no. 2783 in LNCS, pp. 21–26.

[WR02] Wang, C.J.; Resta, P.E.: *Multiplicity & Flexibility as Design and Implementation Features – A Case Study of a Web-Based CL Community for Diverse Learners.* In: Stahl [Sta02], pp. 575–576.

[Yee02] Yee, K.P.: *User Interaction Design for Secure Systems.* In: R. Deng; et al. (eds.), *ICICS 2002*, Springer-Verlag, Berlin Heidelberg, 2002, no. 2513 in LNCS, pp. 278–290.

[Zeu08] Zeuner, P.: *An exemplary modular E-Learning Architecture using the Open Knowledge Initiatives Open Service Interfaces Framework.* Master's thesis, University of Siegen, Oct. 2008.

Appendix A

Complete Fault Tree Analysis Results

A.1 Overview

This appendix illustrates the fault tree as a result of the threat analysis process. It represents the core contribution of this research project. The entire fault tree in a fully expanded version, where all transfer symbols as place holders are actually replaced by their respective subtrees, consists of

- 1085 AND gates,
- 1830 OR gates,
- 4495 basic events, and
- 2915 interim events

⇒ sum of all events: 7410.

As described in Chapter 4, many of these events are included redundantly because of their general character, e.g., threats to the base system components affecting all assets in comparable manner. With neglecting redundant inclusions by not following transfer symbols and only counting unique events and symbols, this results in

- 177 AND gates,
- 276 OR gates,
- 597 basic events, and
- 453 interim events

⇒ sum of all events: 1050.

In the following, the fault tree will be presented similarly to a world map. First, an overview of the entire structure with all subtrees will be given in Figure A.1. Second, as next zoom level, three subtrees will be shown representing the asset groups "e-learning", "information security services", and "web services" as listed in Table 3.4 (cf. Chap. 3, p. 47). Third, after these overviews, more detailed representations will be given in Sections A.2.1–A.2.3 with a zoom level providing human-readable contents. In addition to the main components of the fault tree, in Section A.3, seventeen redundant subtrees will be presented. For the sake of readability, each transfer symbol in the detailed tree presentations is supplemented with references to corresponding figures containing the transfer counter part. Therefore, the reader similarly to a street map will be guided to the next relevant piece of information to continue.

Note that, although for the threat analysis thematically grouped assets were used as starting points, during the actual analysis process, possible reasons for undesired events get mixed, and no clear separation in organisational and technical elements can be assured any more. Obviously, there exist mutual influences among organisational and technical aspects for security. Hence, their placement in different subtrees is required independently of initial asset groups. This means, the deeper events are placed in the tree, the less specific they will probably be with respect to the root of a corresponding subtree.

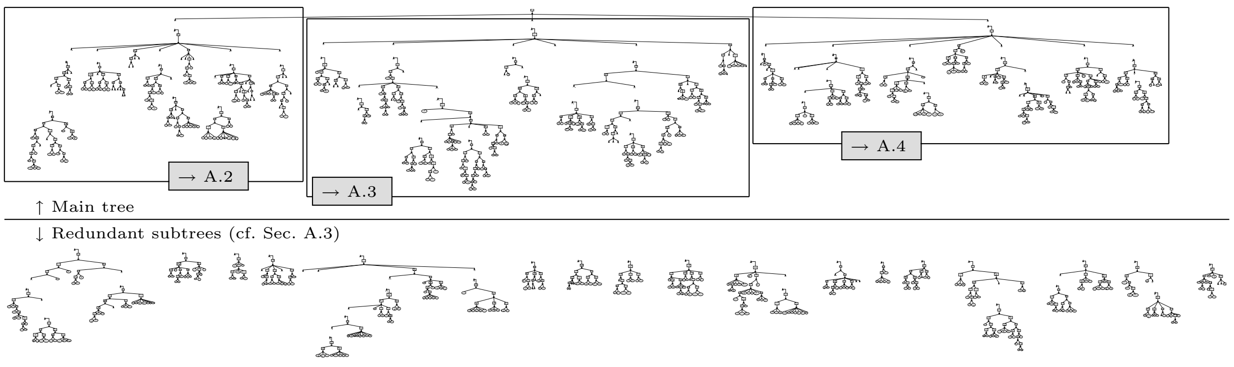


Figure A.1: Overview of all subtrees of the resulting fault tree

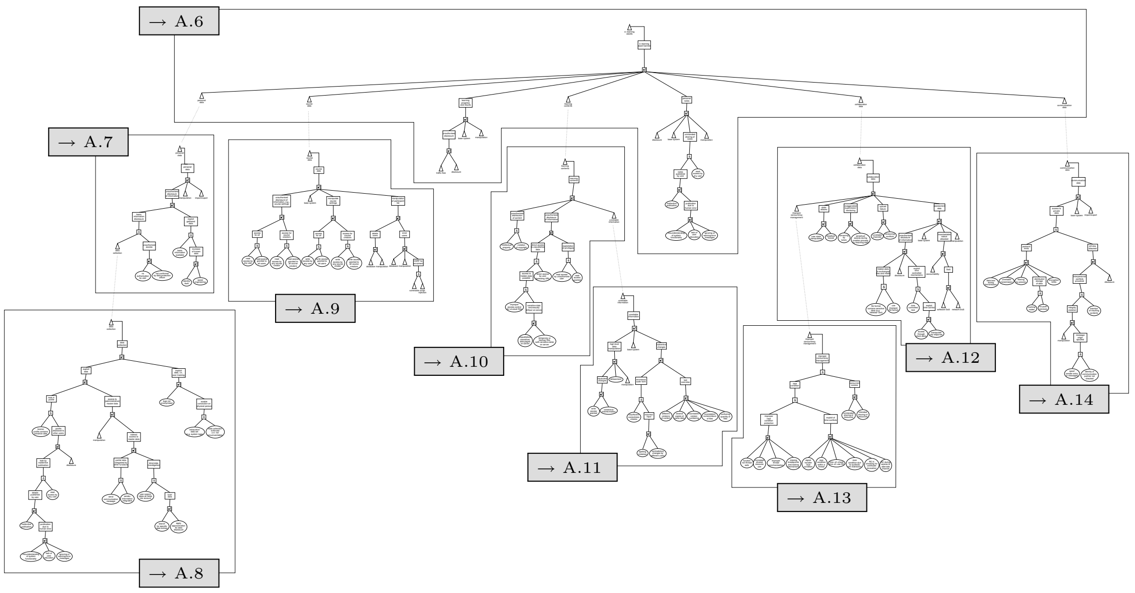


Figure A.2: Overview of the asset group "e-learning"

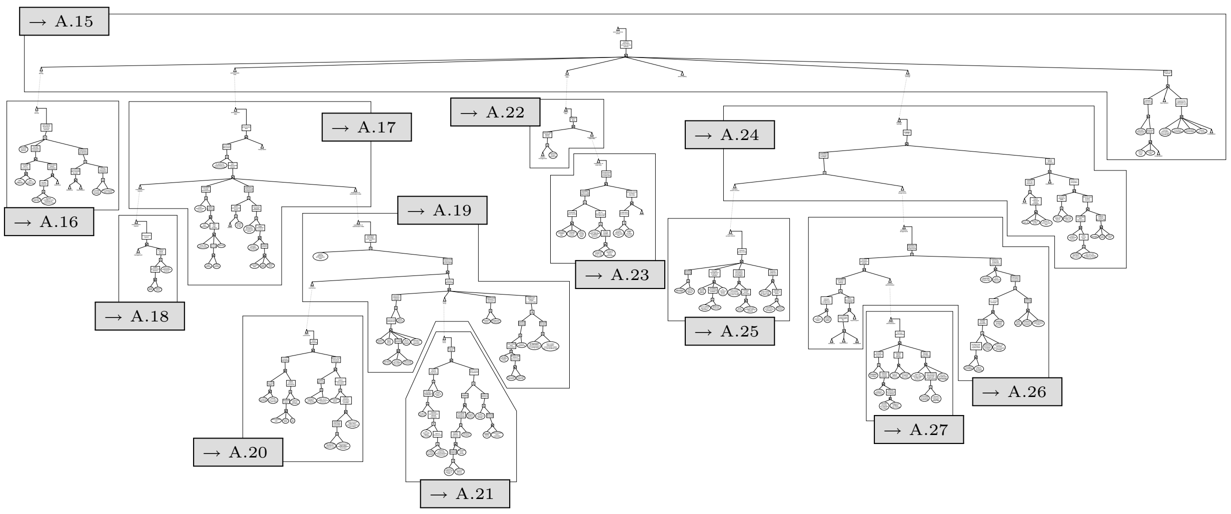


Figure A.3: Overview of asset group "information security services"

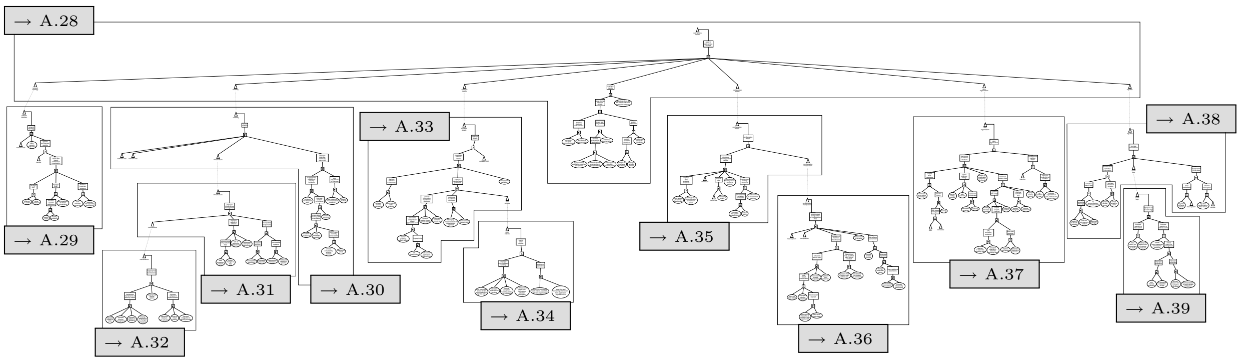


Figure A.4: Overview of asset group "web services"

A.2 Main Tree

The main fault tree consists of the global root element and three connected subtrees that relate to the asset groups as introduced in Chapter 3. These subtree blocks are presented in the following subsections.

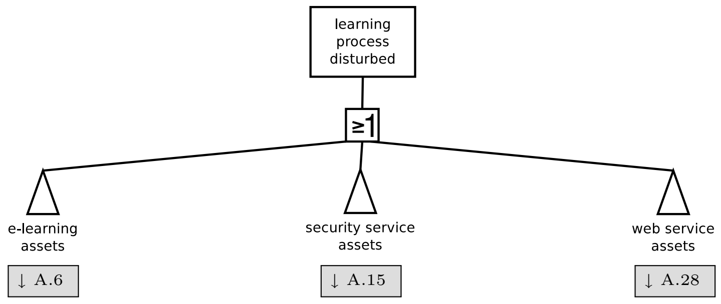


Figure A.5: Root element of fault tree, i.e., learning process disturbed

A.2.1 Block 1: E-Learning Assets

In this first block, i.e., the subtree linking assets of the first asset group, educational contents and learning activities are considered. Due to the amount of specific events and related concepts affecting events in an upper level of the tree, many transfer symbols were necessary to fit the tree on the given page size. Directly related subtrees are presented in this subsection.

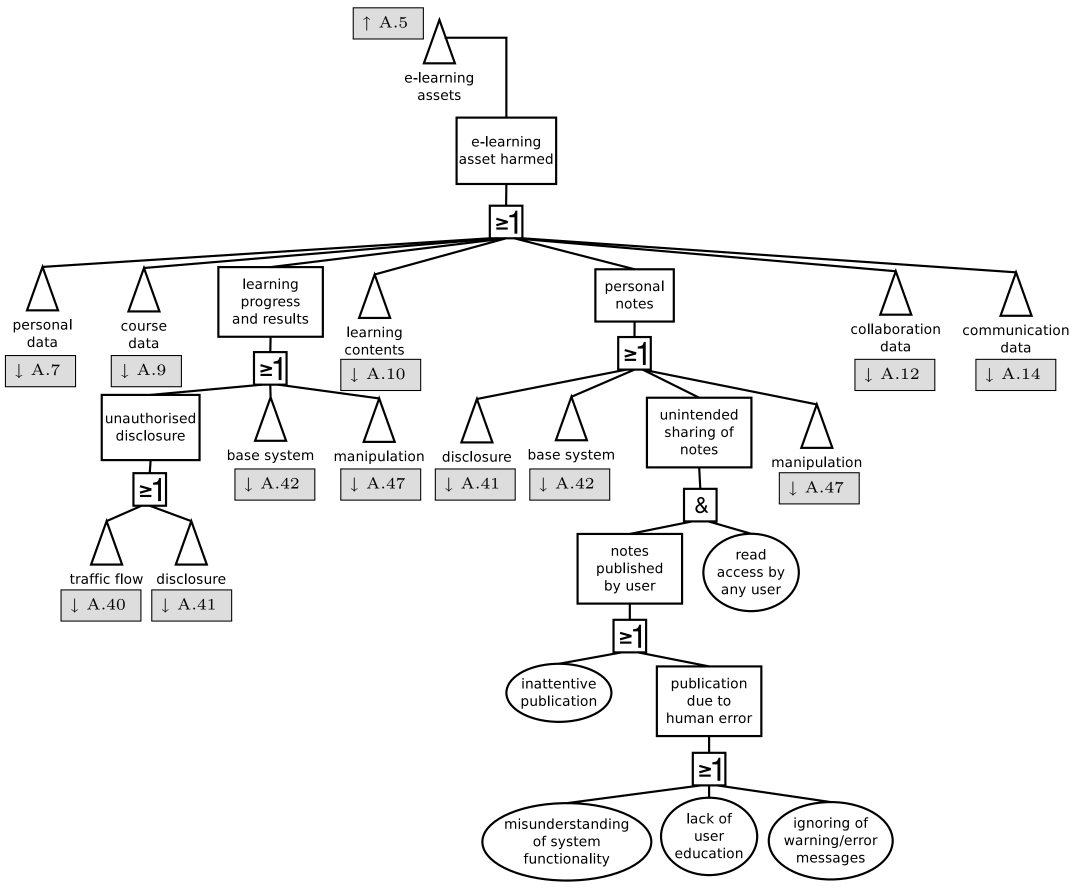


Figure A.6: E-learning asset harmed

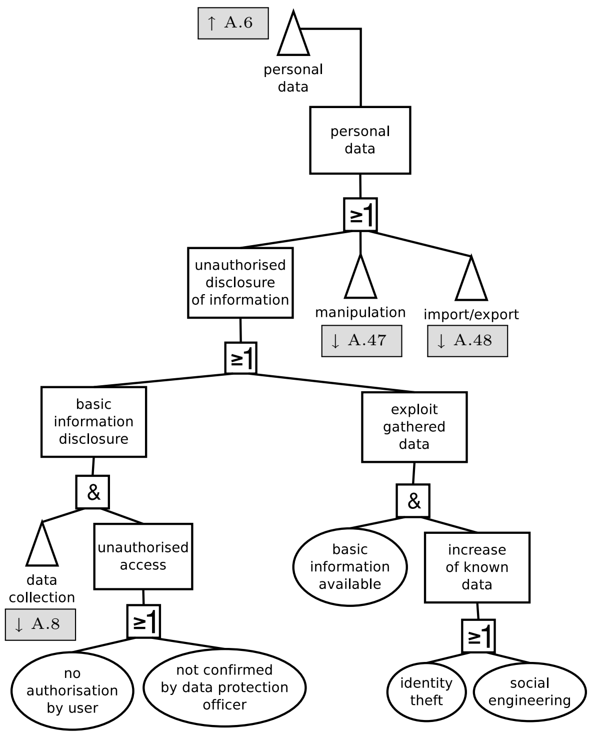


Figure A.7: Person related data

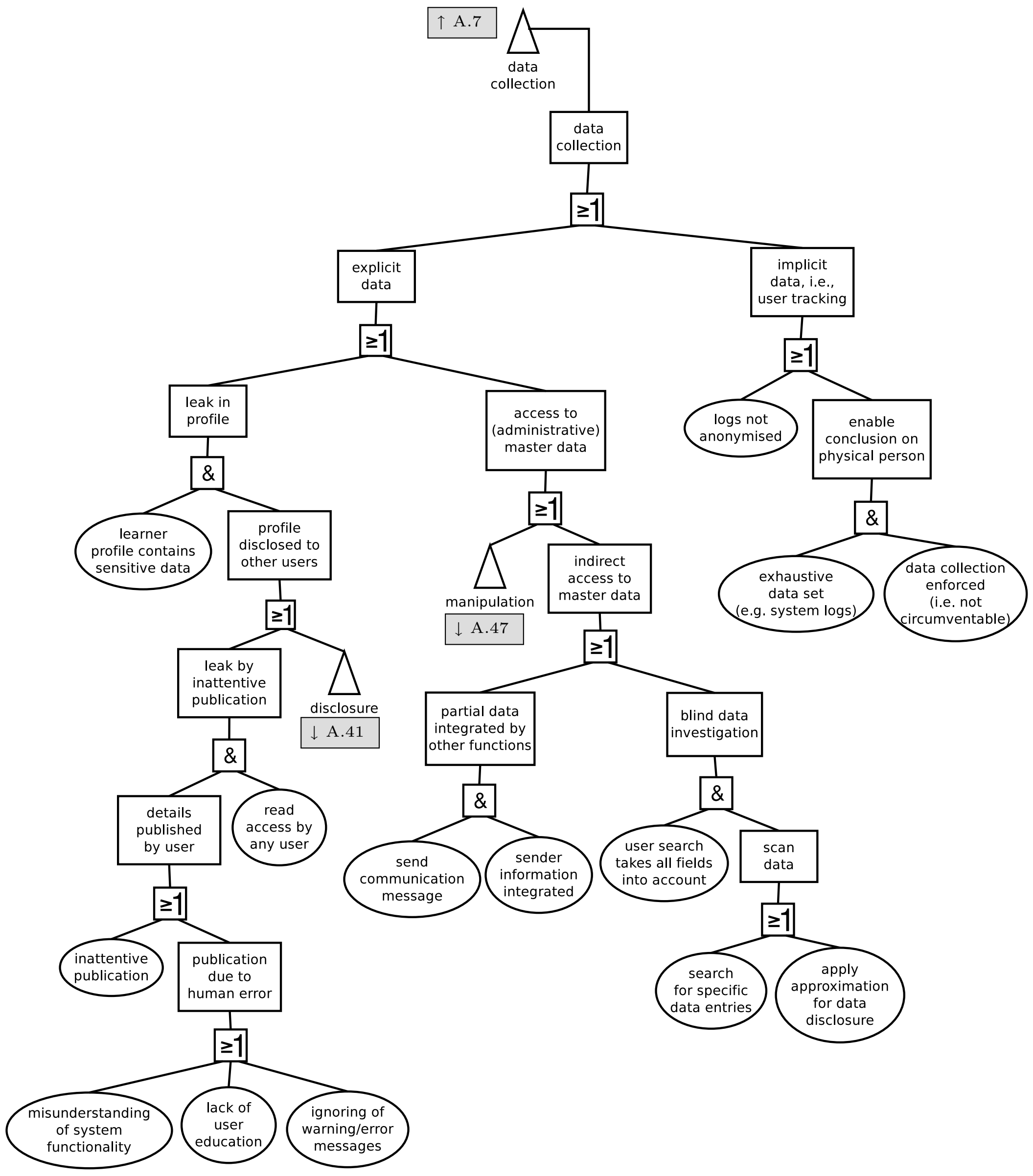

↑ A.7
data collection
data collection
≥1
explicit data
implicit data, i.e., user tracking
≥1
leak in profile
access to (administrative) master data
≥1
logs not anonymised
enable conclusion on physical person
&
learner profile contains sensitive data
profile disclosed to other users
≥1
manipulation
↓ A.47
indirect access to master data
&
exhaustive data set (e.g. system logs)
data collection enforced (i.e. not circumventable)
≥1
leak by inattentive publication
disclosure
↓ A.41
≥1
partial data integrated by other functions
blind data investigation
&
details published by user
read access by any user
&
send communication message
sender information integrated
&
user search takes all fields into account
scan data
≥1
inattentive publication
publication due to human error
≥1
search for specific data entries
apply approximation for data disclosure
≥1
misunderstanding of system functionality
lack of user education
ignoring of warning/error messages

Figure A.8: Data collection

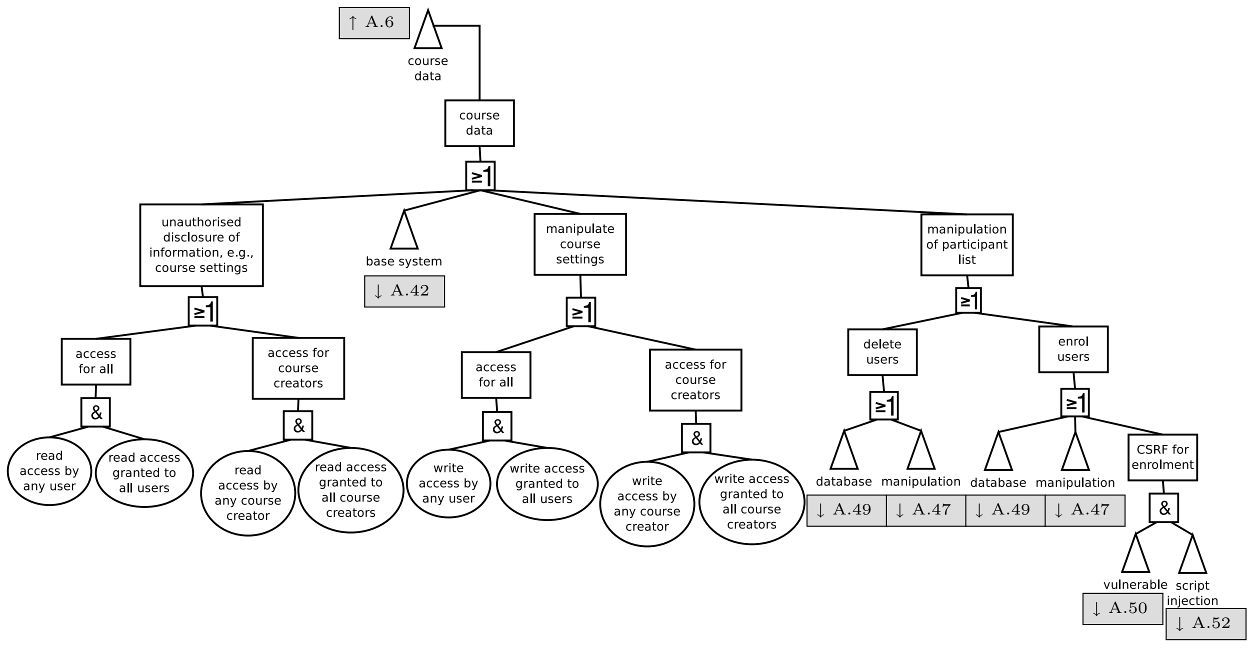

↑ A.6
course data
course data
≥1
unauthorised disclosure of information, e.g., course settings
base system
↓ A.42
manipulate course settings
manipulation of participant list
≥1
access for all
access for course creators
&
read access by any user
read access granted to all users
&
read access by any course creator
read access granted to all course creators
≥1
access for all
access for course creators
&
write access by any user
write access granted to all users
&
write access by any course creator
write access granted to all course creators
≥1
delete users
enrol users
≥1
database
manipulation
↓ A.49
↓ A.47
≥1
database
manipulation
↓ A.49
↓ A.47
CSRF for enrolment
&
vulnerable
↓ A.50
script injection
↓ A.52

Figure A.9: Course data

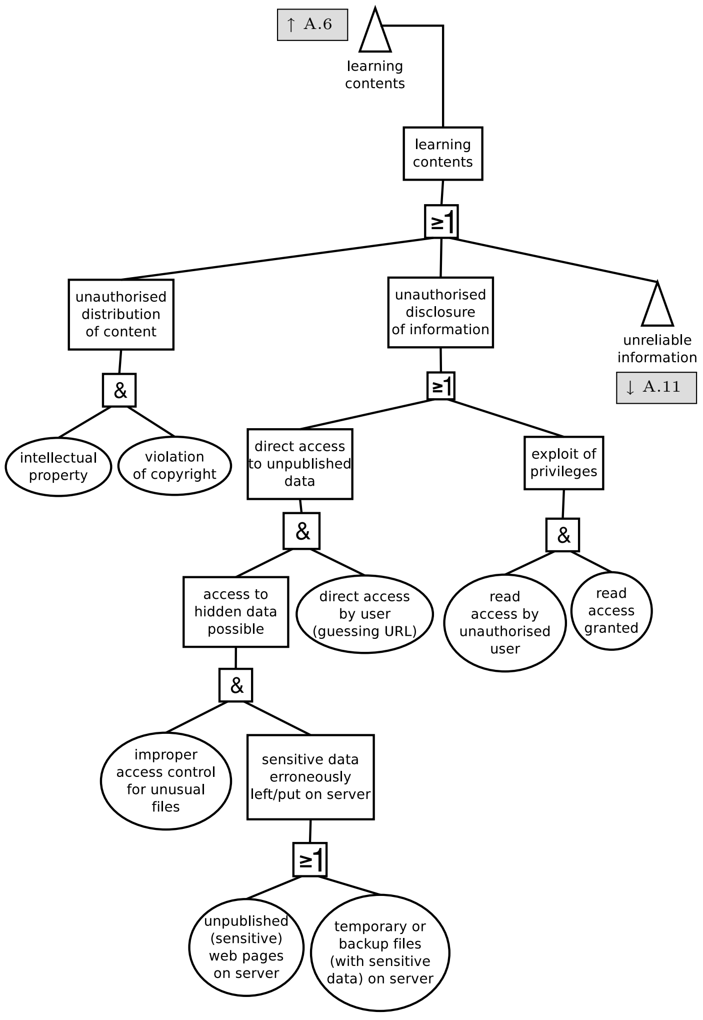

↑ A.6
learning contents
learning contents
≥1
unauthorised distribution of content
unauthorised disclosure of information
unreliable information
↓ A.11
&
intellectual property
violation of copyright
≥1
direct access to unpublished data
exploit of privileges
&
access to hidden data possible
direct access by user (guessing URL)
&
read access by unauthorised user
read access granted
&
improper access control for unusual files
sensitive data erroneously left/put on server
≥1
unpublished (sensitive) web pages on server
temporary or backup files (with sensitive data) on server

Figure A.10: Learning contents

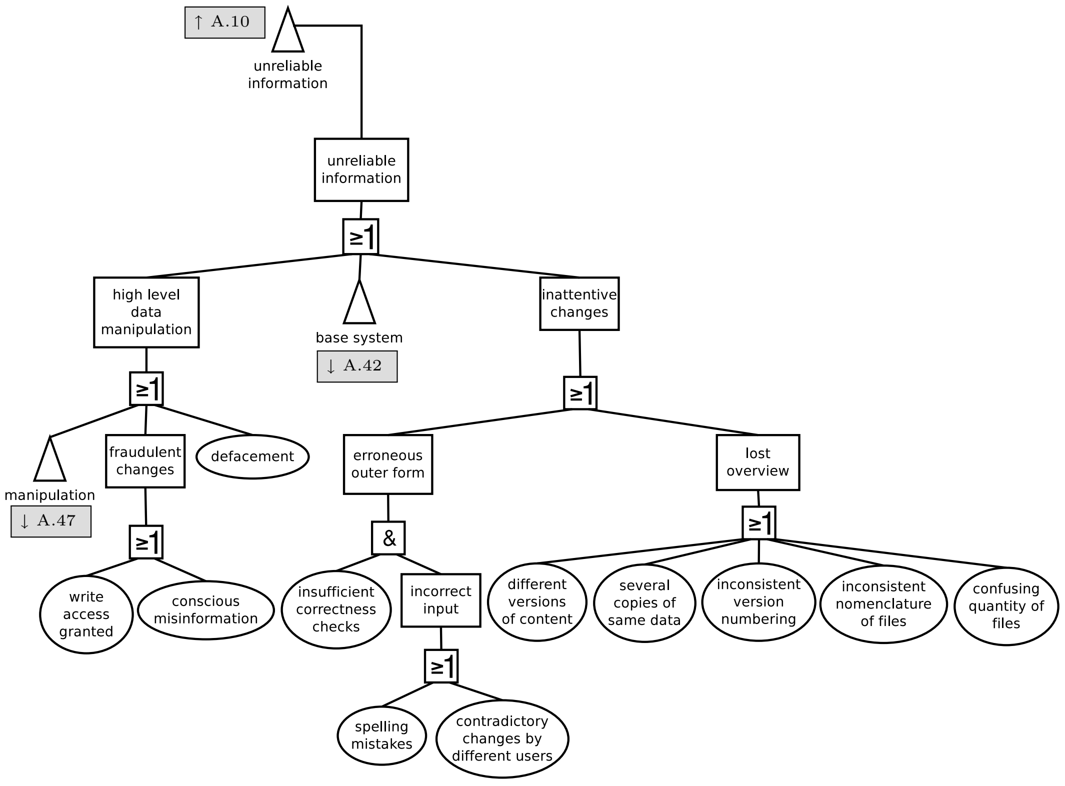


Figure A.11: Unreliable information

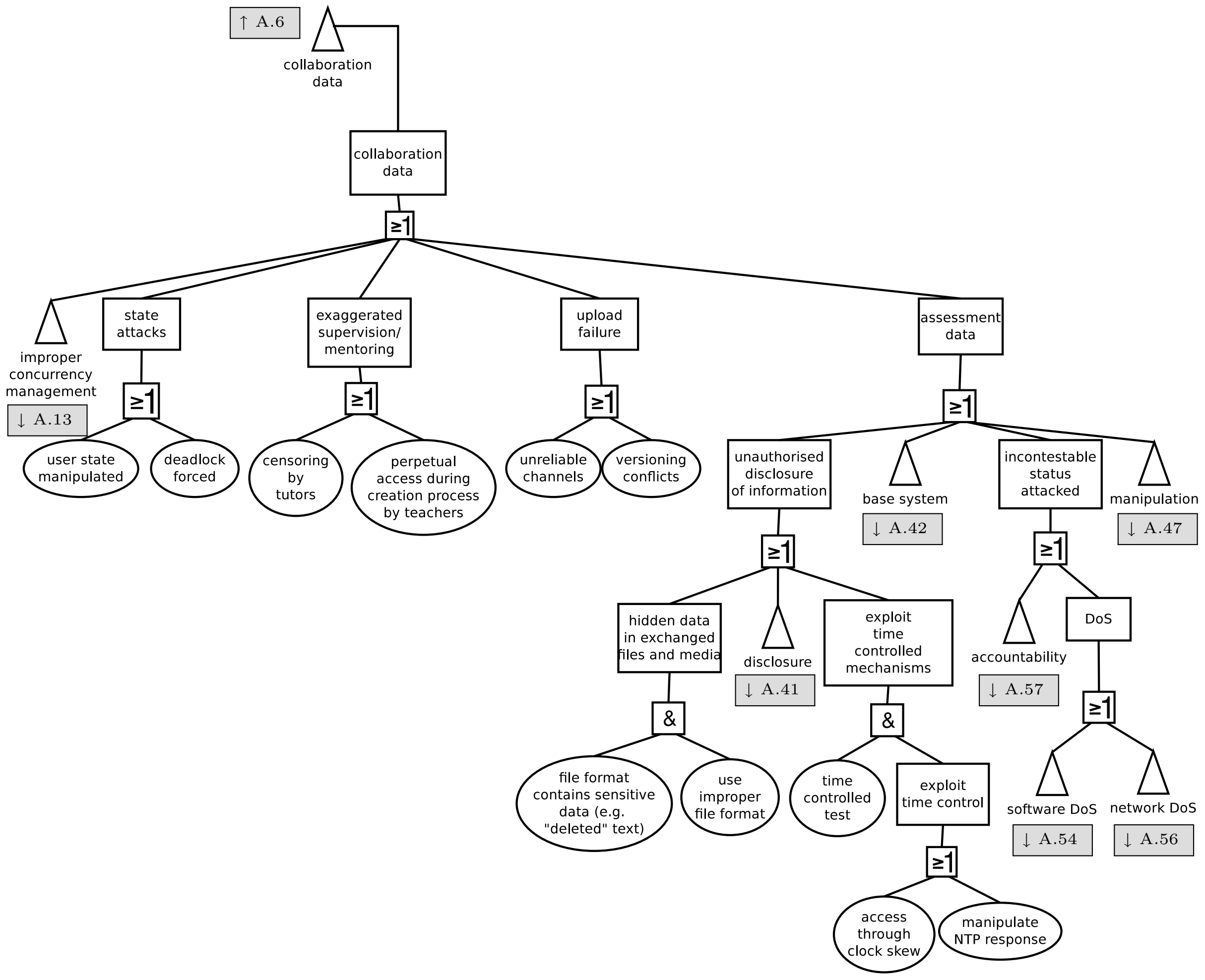


Figure A.12: Collaboration data

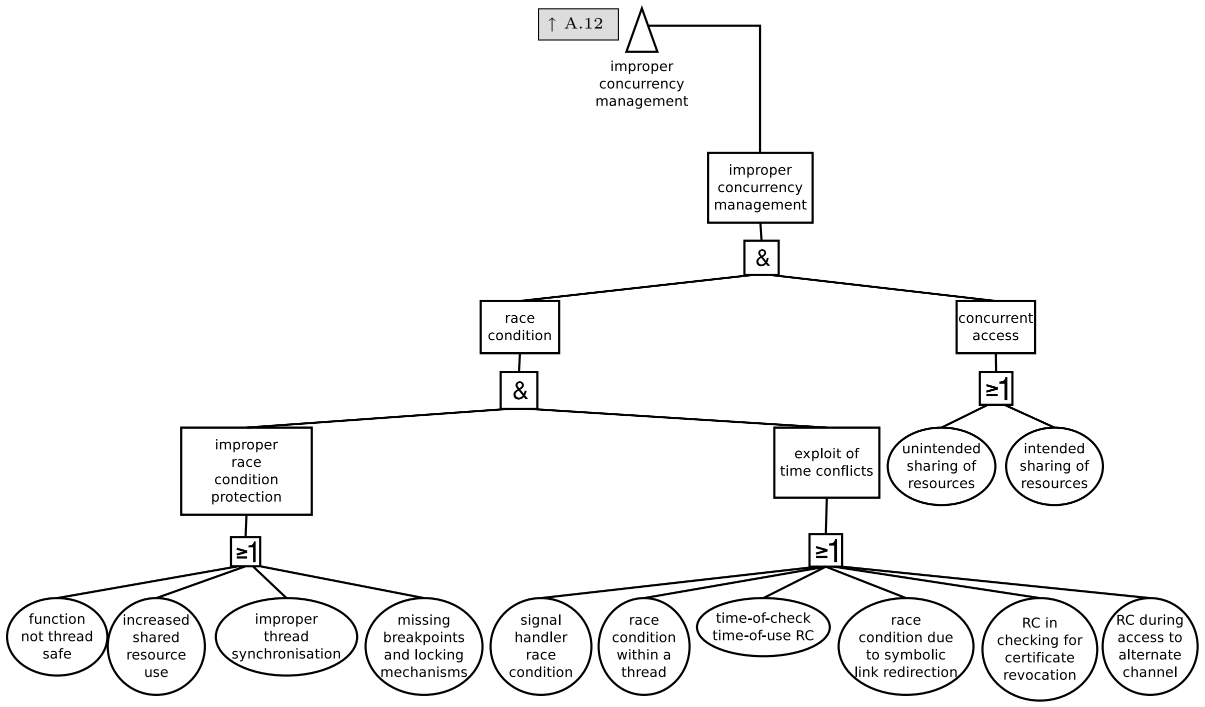

↑ A.12
improper concurrency management
improper concurrency management
&
race condition
&
improper race condition protection
≥1
function not thread safe
increased shared resource use
improper thread synchronisation
missing breakpoints and locking mechanisms
exploit of time conflicts
≥1
signal handler race condition
race condition within a thread
time-of-check time-of-use RC
race condition due to symbolic link redirection
RC in checking for certificate revocation
RC during access to alternate channel
concurrent access
≥1
unintended sharing of resources
intended sharing of resources

Figure A.13: Improper concurrency management

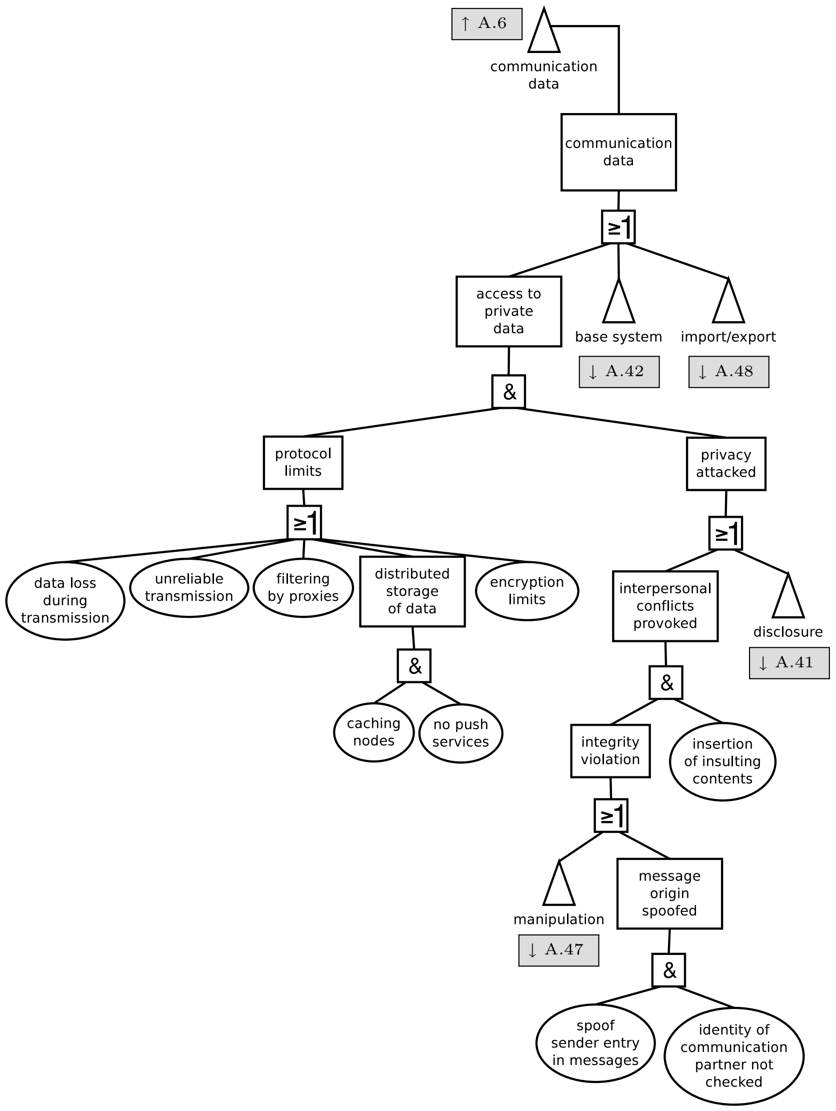


Figure A.14: Communication data

A.2.2 Block 2: Information Security Service Assets

In the second block, assets connected to information security services are considered. This primarily concerns policies, authentication, and privilege settings. Related subtrees are given in the following and marked by references to support the reader in following paths.

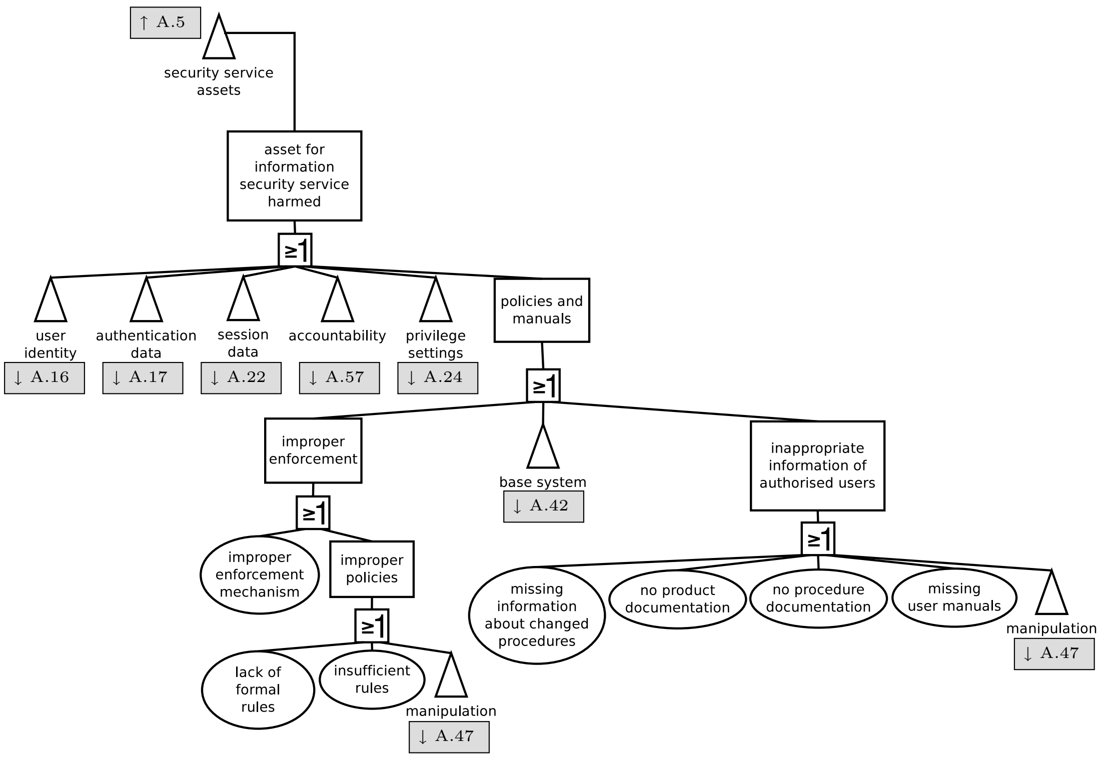


Figure A.15: Asset for information security services harmed

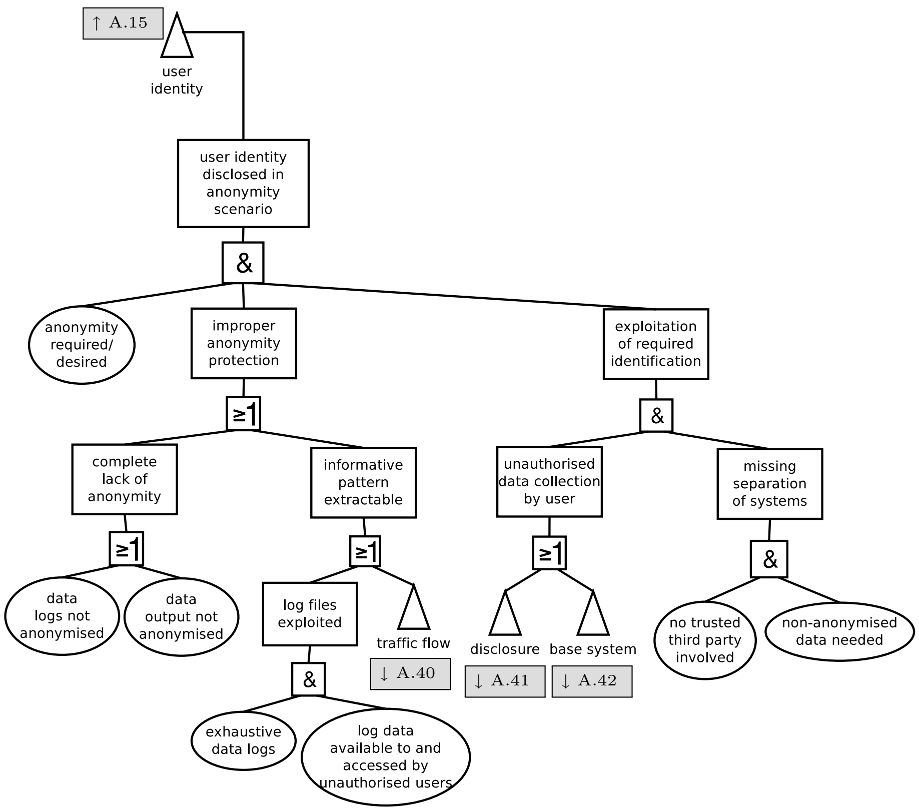

↑ A.15
user identity
user identity disclosed in anonymity scenario
&
anonymity required/ desired
improper anonymity protection
exploitation of required identification
≥1
&
complete lack of anonymity
informative pattern extractable
unauthorised data collection by user
missing separation of systems
≥1
≥1
≥1
&
data logs not anonymised
data output not anonymised
log files exploited
traffic flow
↓ A.40
disclosure
base system
↓ A.41
↓ A.42
no trusted third party involved
non-anonymised data needed
&
exhaustive data logs
log data available to and accessed by unauthorised users

Figure A.16: User identity disclosed in anonymity scenario

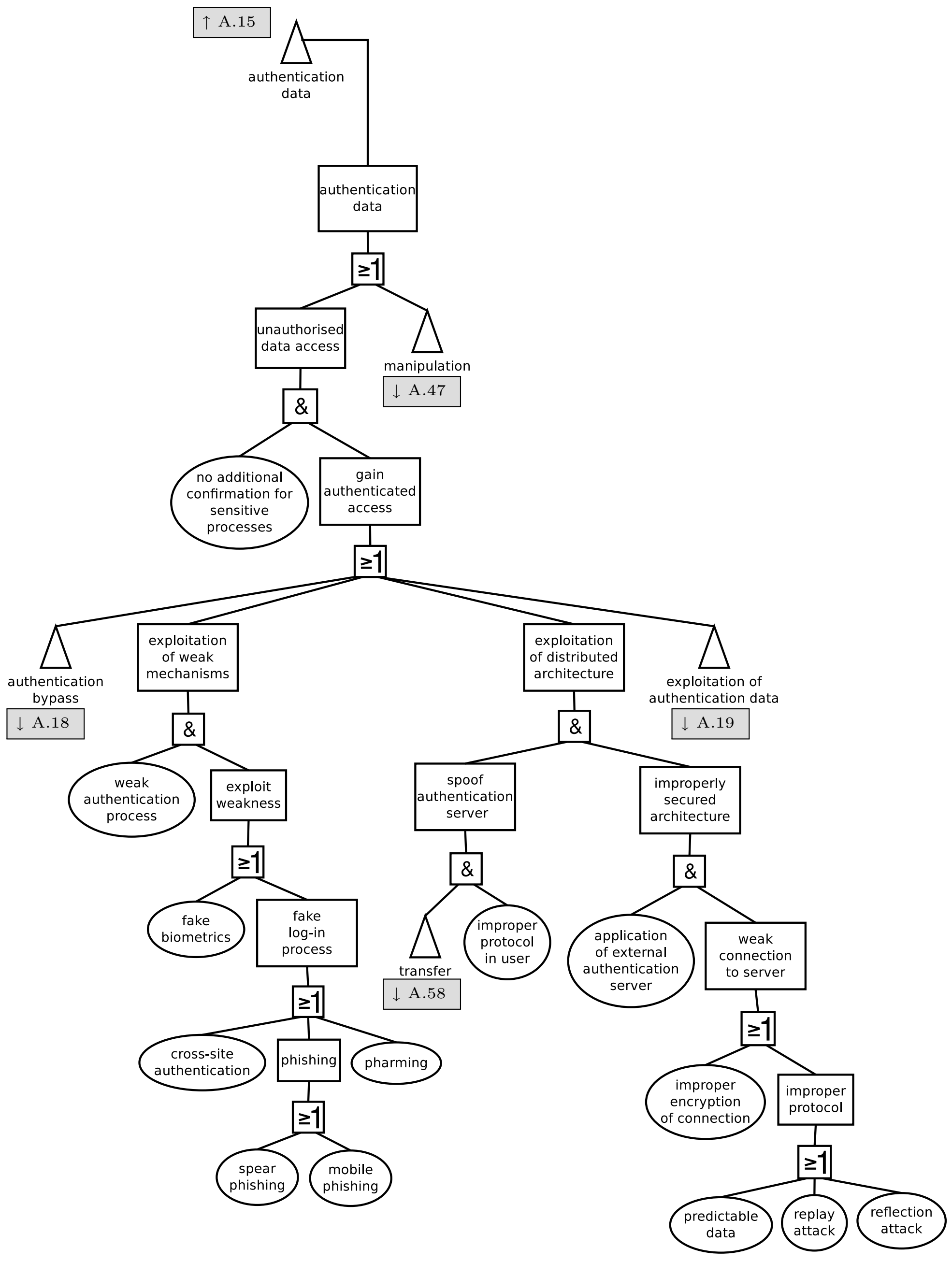

↑ A.15
authentication data
authentication data
≥1
unauthorised data access
manipulation
↓ A.47
&
no additional confirmation for sensitive processes
gain authenticated access
≥1
authentication bypass
↓ A.18
exploitation of weak mechanisms
exploitation of distributed architecture
exploitation of authentication data
↓ A.19
&
weak authentication process
exploit weakness
≥1
fake biometrics
fake log-in process
≥1
cross-site authentication
phishing
pharming
≥1
spear phishing
mobile phishing
&
spoof authentication server
improperly secured architecture
&
transfer
↓ A.58
improper protocol in user
&
application of external authentication server
weak connection to server
≥1
improper encryption of connection
improper protocol
≥1
predictable data
replay attack
reflection attack

Figure A.17: Authentication data

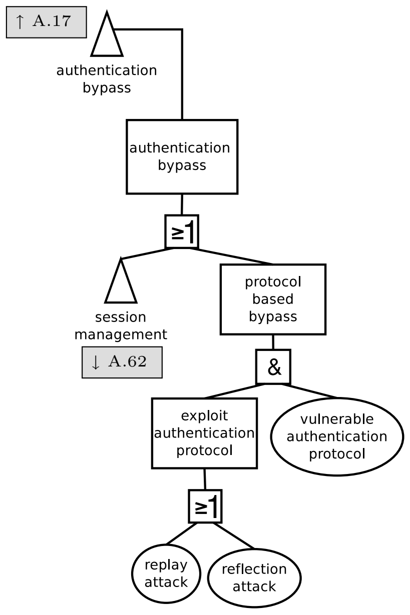


Figure A.18: Authentication bypass

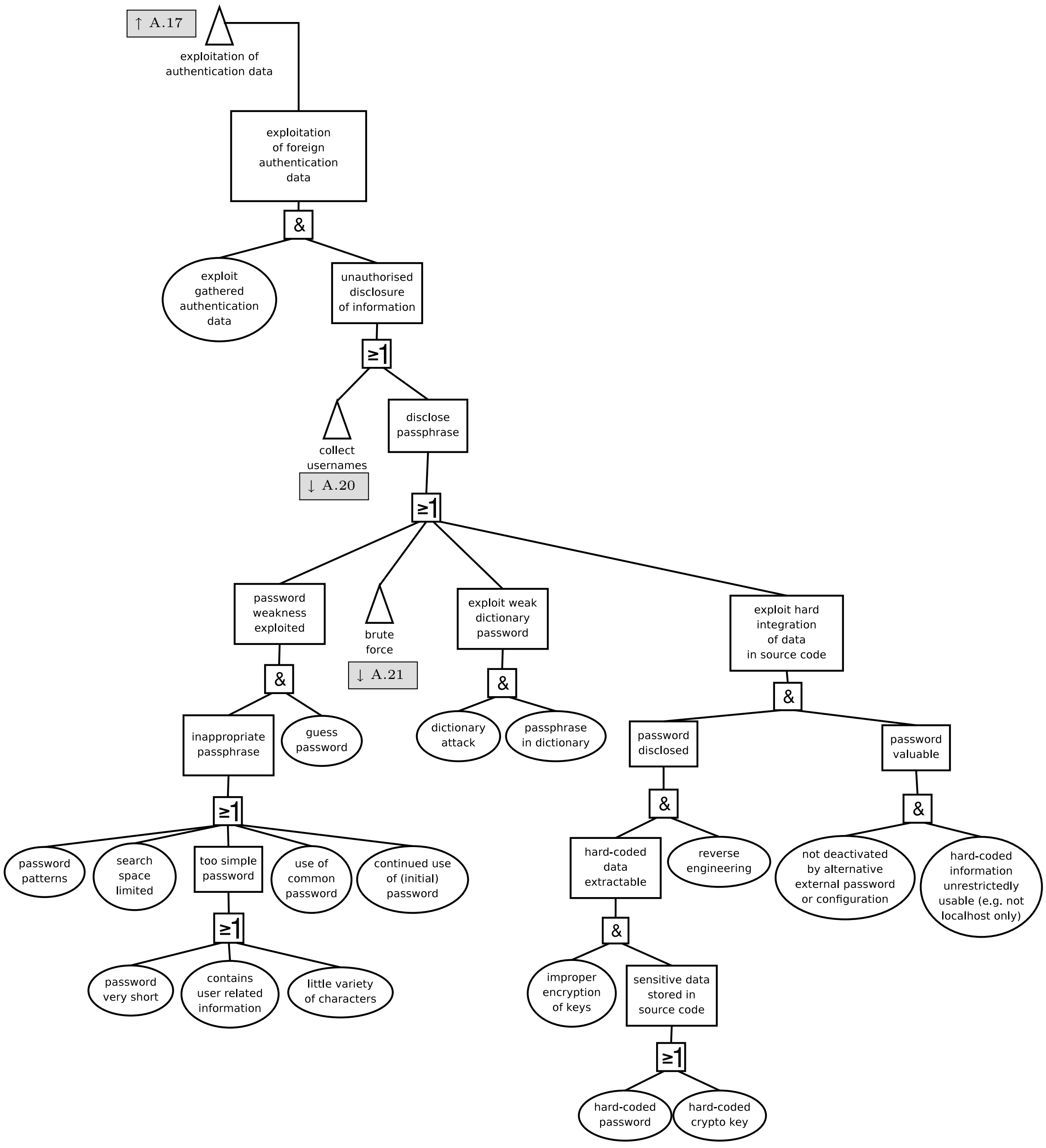

↑ A.17
exploitation of authentication data
exploitation of foreign authentication data
&
exploit gathered authentication data
unauthorised disclosure of information
≥1
collect usernames
↓ A.20
disclose passphrase
≥1
password weakness exploited
brute force
↓ A.21
exploit weak dictionary password
exploit hard integration of data in source code
&
inappropriate passphrase
guess password
&
dictionary attack
passphrase in dictionary
&
password disclosed
password valuable
≥1
password patterns
search space limited
too simple password
use of common password
continued use of (initial) password
≥1
password very short
contains user related information
little variety of characters
&
hard-coded data extractable
reverse engineering
&
not deactivated by alternative external password or configuration
hard-coded information unrestrictedly usable (e.g. not localhost only)
&
improper encryption of keys
sensitive data stored in source code
≥1
hard-coded password
hard-coded crypto key

Figure A.19: Exploitation of foreign authentication data

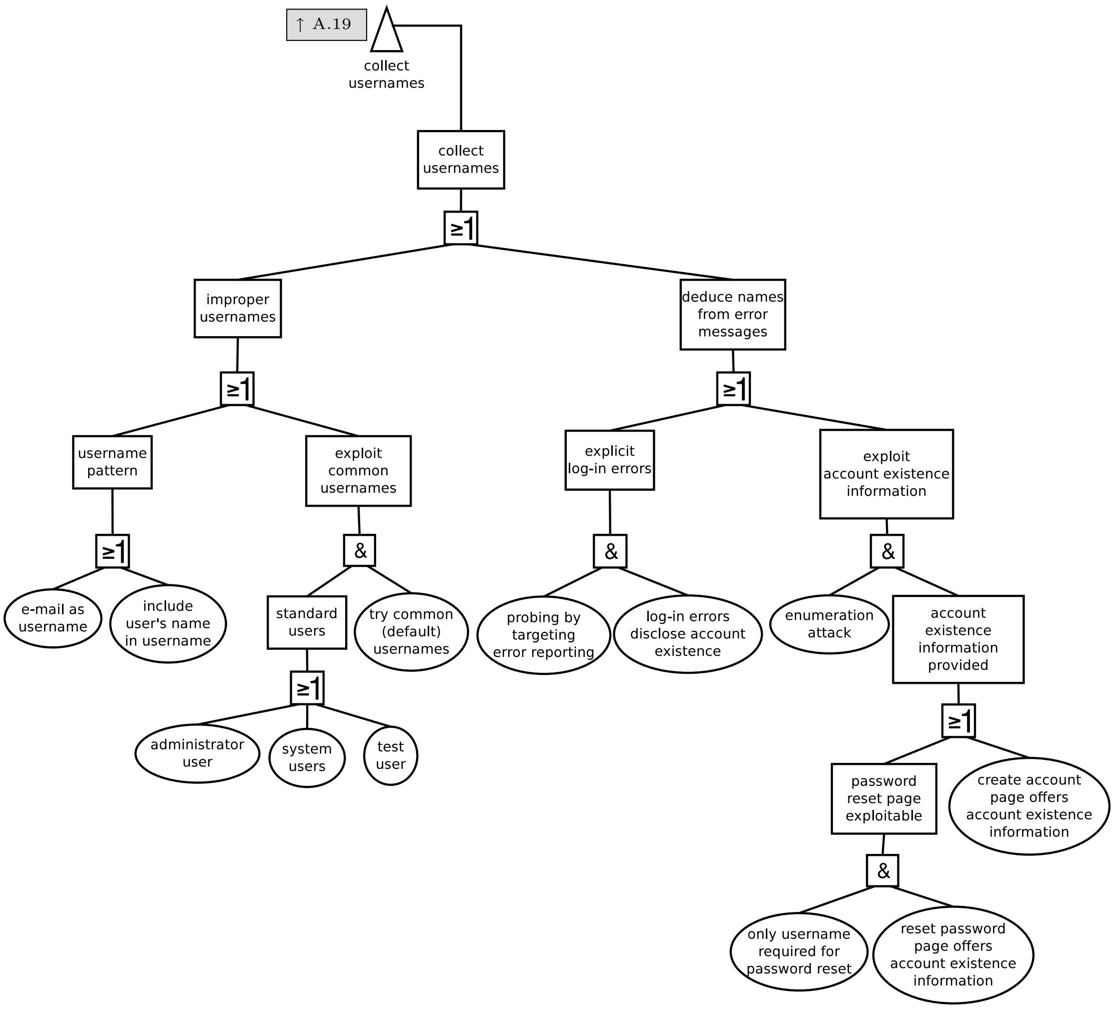

↑ A.19
collect usernames
collect usernames
≥1
improper usernames
deduce names from error messages
≥1
username pattern
exploit common usernames
≥1
e-mail as username
include user's name in username
&
standard users
try common (default) usernames
≥1
administrator user
system users
test user
≥1
explicit log-in errors
exploit account existence information
&
probing by targeting error reporting
log-in errors disclose account existence
&
enumeration attack
account existence information provided
≥1
password reset page exploitable
create account page offers account existence information
&
only username required for password reset
reset password page offers account existence information

Figure A.20: Collect usernames

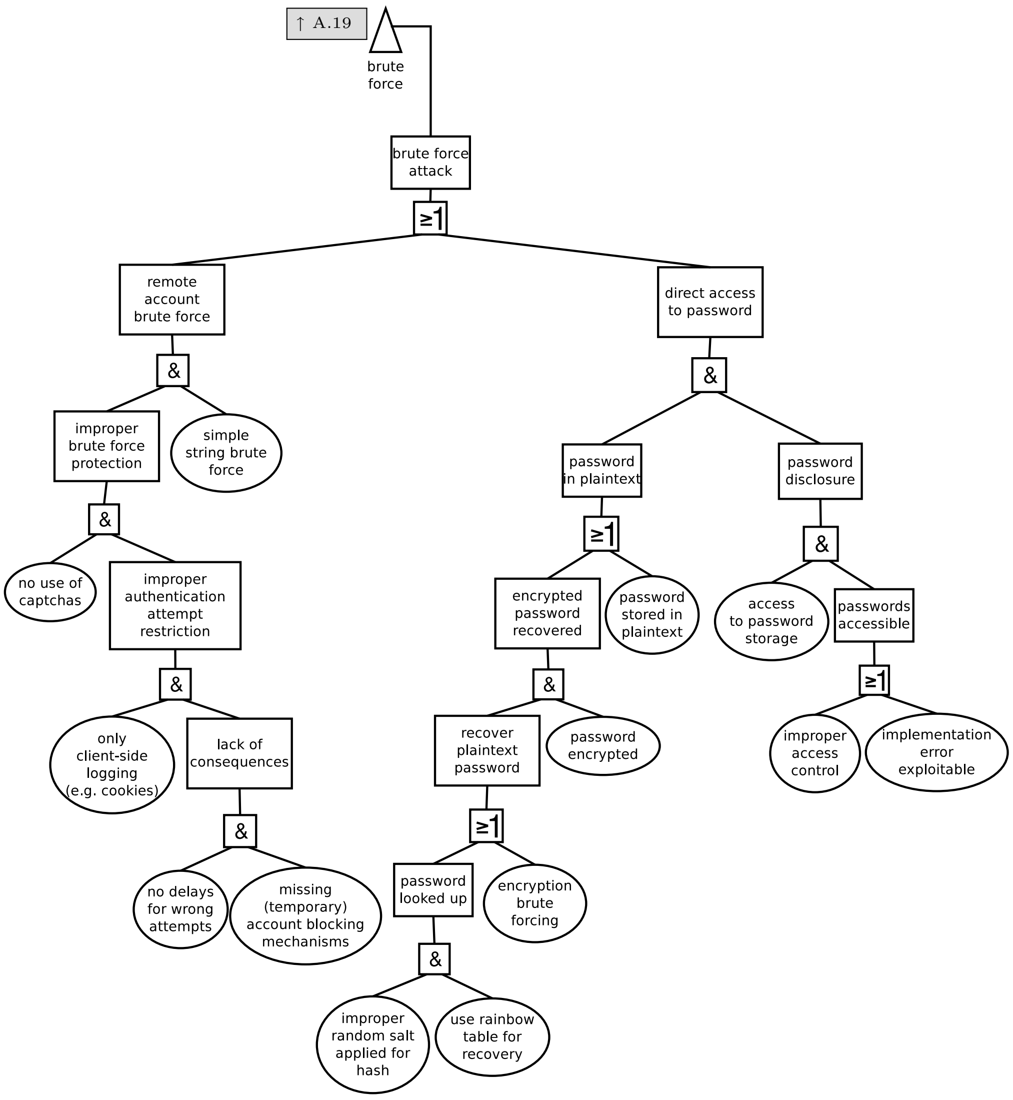

↑ A.19
brute force
brute force attack
≥1
remote account brute force
&
improper brute force protection
simple string brute force
&
no use of captchas
improper authentication attempt restriction
&
only client-side logging (e.g. cookies)
lack of consequences
&
no delays for wrong attempts
missing (temporary) account blocking mechanisms
direct access to password
&
password in plaintext
≥1
encrypted password recovered
password stored in plaintext
&
recover plaintext password
password encrypted
≥1
password looked up
encryption brute forcing
&
improper random salt applied for hash
use rainbow table for recovery
password disclosure
&
access to password storage
passwords accessible
≥1
improper access control
implementation error exploitable

Figure A.21: Brute force attack

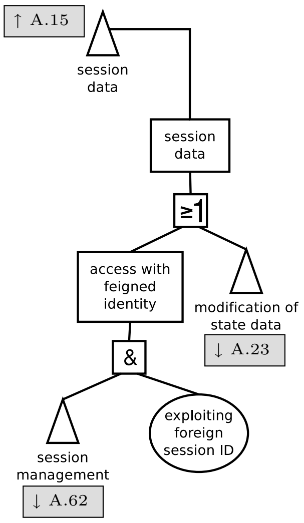

↑ A.15
session data
session data
≥1
access with feigned identity
&
session management
↓ A.62
exploiting foreign session ID
modification of state data
↓ A.23

Figure A.22: Session data

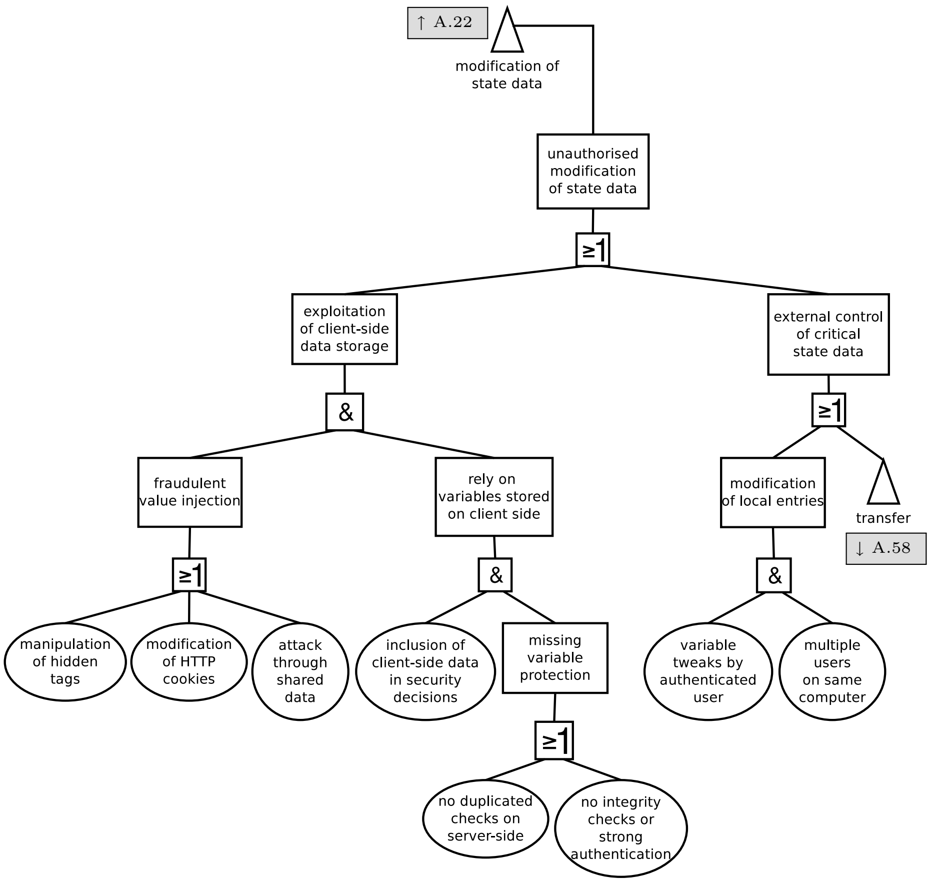

↑ A.22
modification of state data
unauthorised modification of state data
≥1
exploitation of client-side data storage
external control of critical state data
&
fraudulent value injection
rely on variables stored on client side
≥1
modification of local entries
transfer
↓ A.58
≥1
&
&
manipulation of hidden tags
modification of HTTP cookies
attack through shared data
inclusion of client-side data in security decisions
missing variable protection
variable tweaks by authenticated user
multiple users on same computer
≥1
no duplicated checks on server-side
no integrity checks or strong authentication

Figure A.23: Unauthorised modification of state data

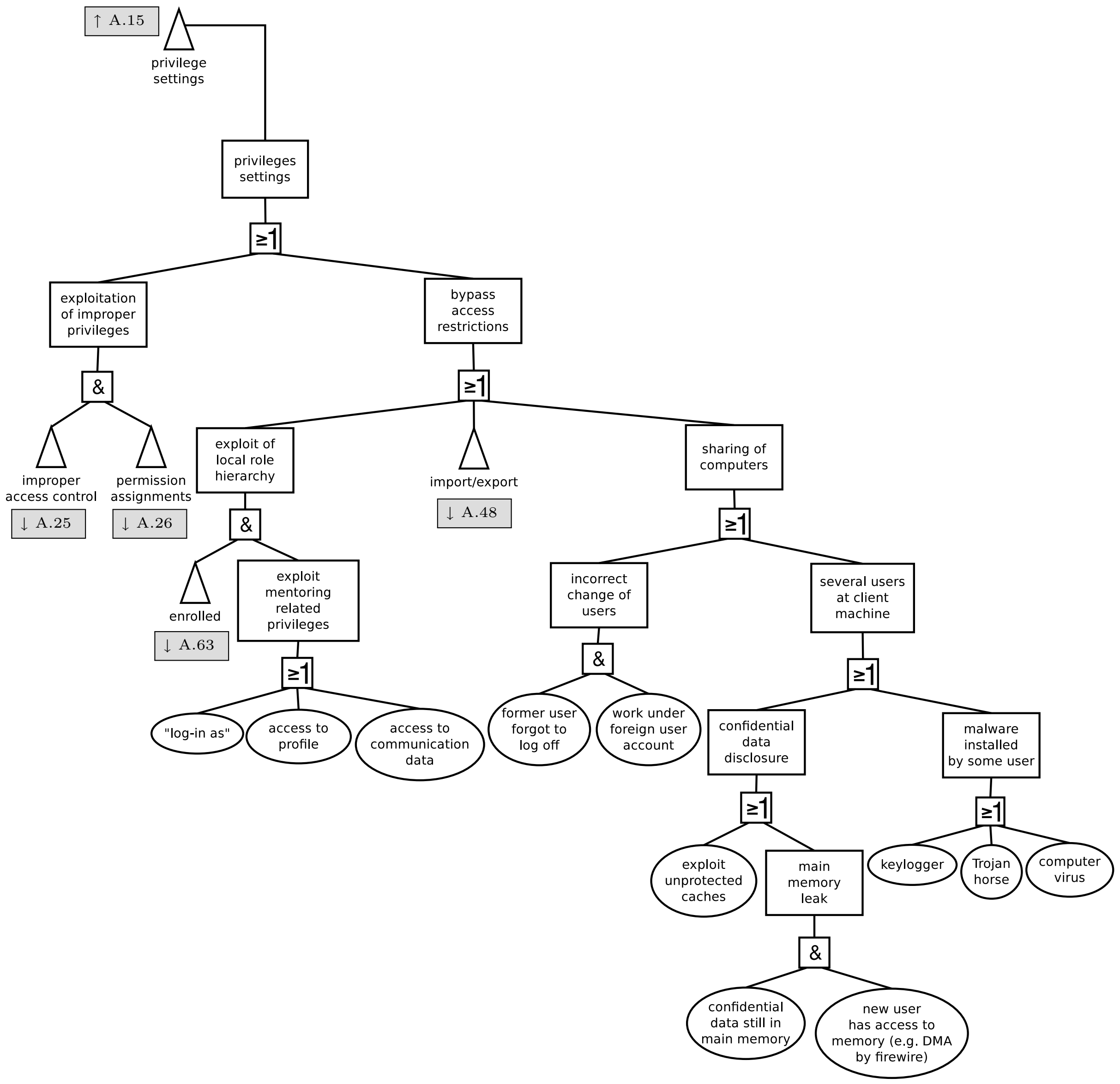

↑ A.15
privilege settings
privileges settings
≥1
exploitation of improper privileges
bypass access restrictions
&
improper access control
↓ A.25
permission assignments
↓ A.26
≥1
exploit of local role hierarchy
import/export
↓ A.48
sharing of computers
&
enrolled
↓ A.63
exploit mentoring related privileges
≥1
"log-in as"
access to profile
access to communication data
≥1
incorrect change of users
&
former user forgot to log off
work under foreign user account
several users at client machine
≥1
confidential data disclosure
≥1
exploit unprotected caches
main memory leak
&
confidential data still in main memory
new user has access to memory (e.g. DMA by firewire)
malware installed by some user
≥1
keylogger
Trojan horse
computer virus

Figure A.24: Privilege settings

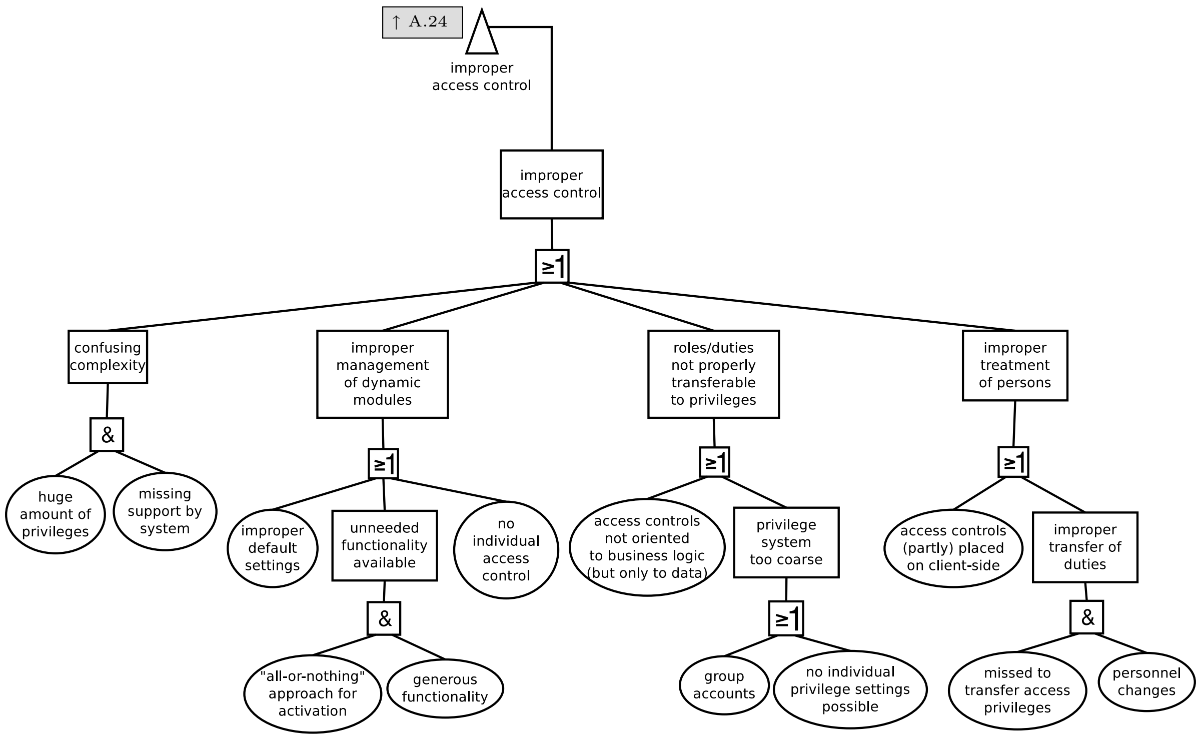

↑ A.24
improper access control
improper access control
≥1
confusing complexity
&
huge amount of privileges
missing support by system
improper management of dynamic modules
≥1
improper default settings
unneeded functionality available
&
"all-or-nothing" approach for activation
generous functionality
no individual access control
roles/duties not properly transferable to privileges
≥1
access controls not oriented to business logic (but only to data)
privilege system too coarse
≥1
group accounts
no individual privilege settings possible
improper treatment of persons
≥1
access controls (partly) placed on client-side
improper transfer of duties
&
missed to transfer access privileges
personnel changes

Figure A.25: Improper access control

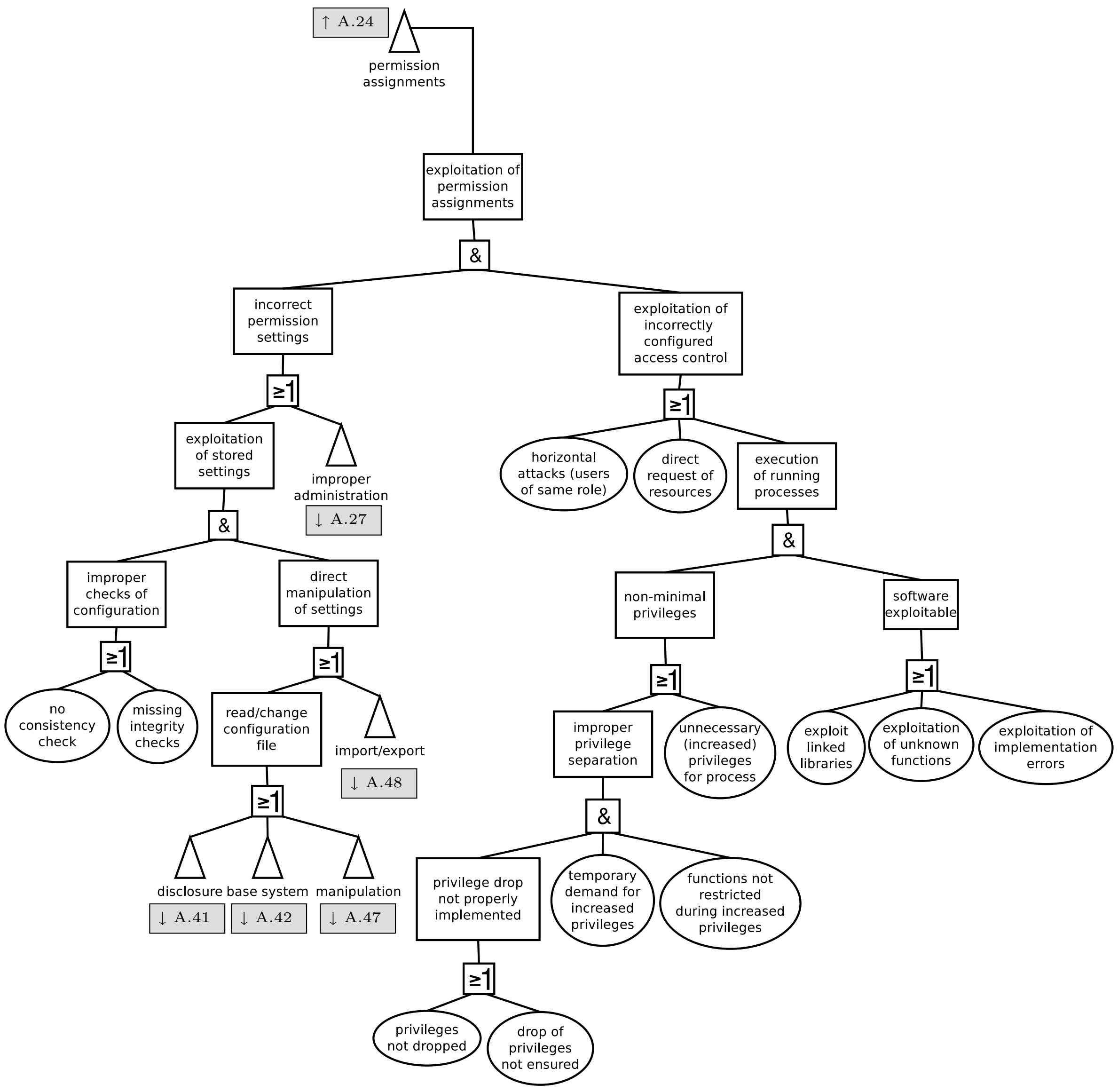

↑ A.24
permission assignments
exploitation of permission assignments
&
incorrect permission settings
exploitation of incorrectly configured access control
≥1
exploitation of stored settings
improper administration
↓ A.27
horizontal attacks (users of same role)
direct request of resources
execution of running processes
&
improper checks of configuration
direct manipulation of settings
non-minimal privileges
software exploitable
no consistency check
missing integrity checks
read/change configuration file
import/export
↓ A.48
improper privilege separation
unnecessary (increased) privileges for process
exploit linked libraries
exploitation of unknown functions
exploitation of implementation errors
disclosure
base system
manipulation
↓ A.41
↓ A.42
↓ A.47
privilege drop not properly implemented
temporary demand for increased privileges
functions not restricted during increased privileges
privileges not dropped
drop of privileges not ensured

Figure A.26: Exploitation of permission assignments

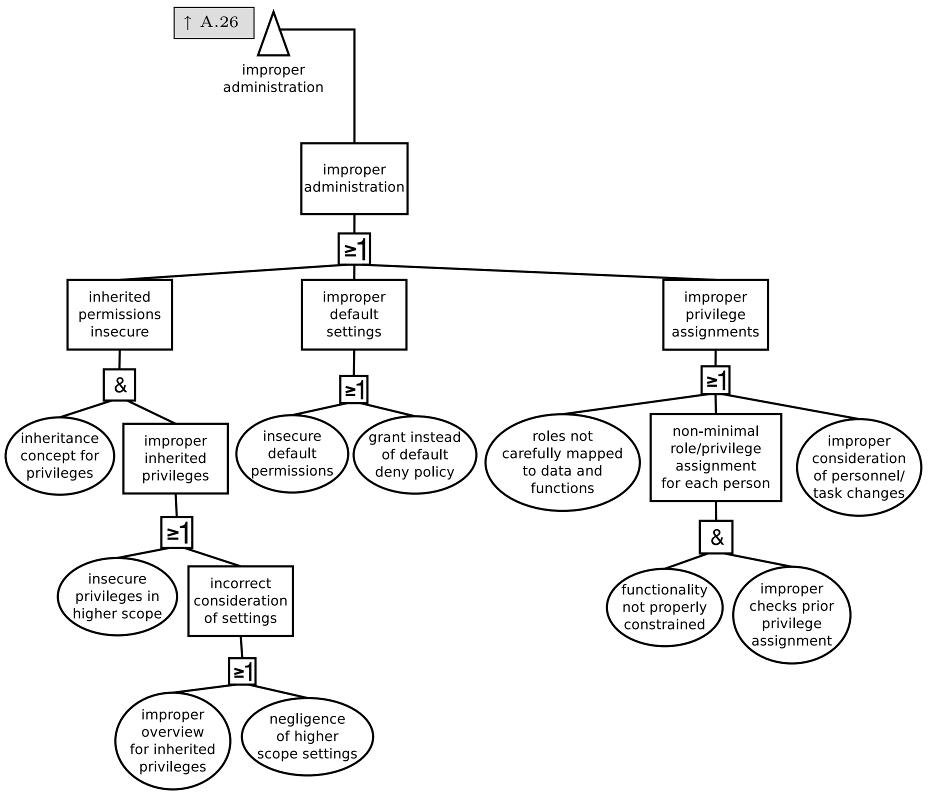

↑ A.26
improper administration
improper administration
≥1
inherited permissions insecure
improper default settings
improper privilege assignments
&
inheritance concept for privileges
improper inherited privileges
≥1
insecure privileges in higher scope
incorrect consideration of settings
≥1
improper overview for inherited privileges
negligence of higher scope settings
≥1
insecure default permissions
grant instead of default deny policy
≥1
roles not carefully mapped to data and functions
non-minimal role/privilege assignment for each person
improper consideration of personnel/ task changes
&
functionality not properly constrained
improper checks prior privilege assignment

Figure A.27: Improper administration

A.2.3 Block 3: Web Services Assets

The third block deals with threats affecting more generic assets inherent in almost all web services. Concerned assets and respective events are less specific to e-learning, but affect basic components. Note that attacks to these components could result in a circumvention of security means within the e-learning system. Consequently, their consideration in this analysis is required as well.

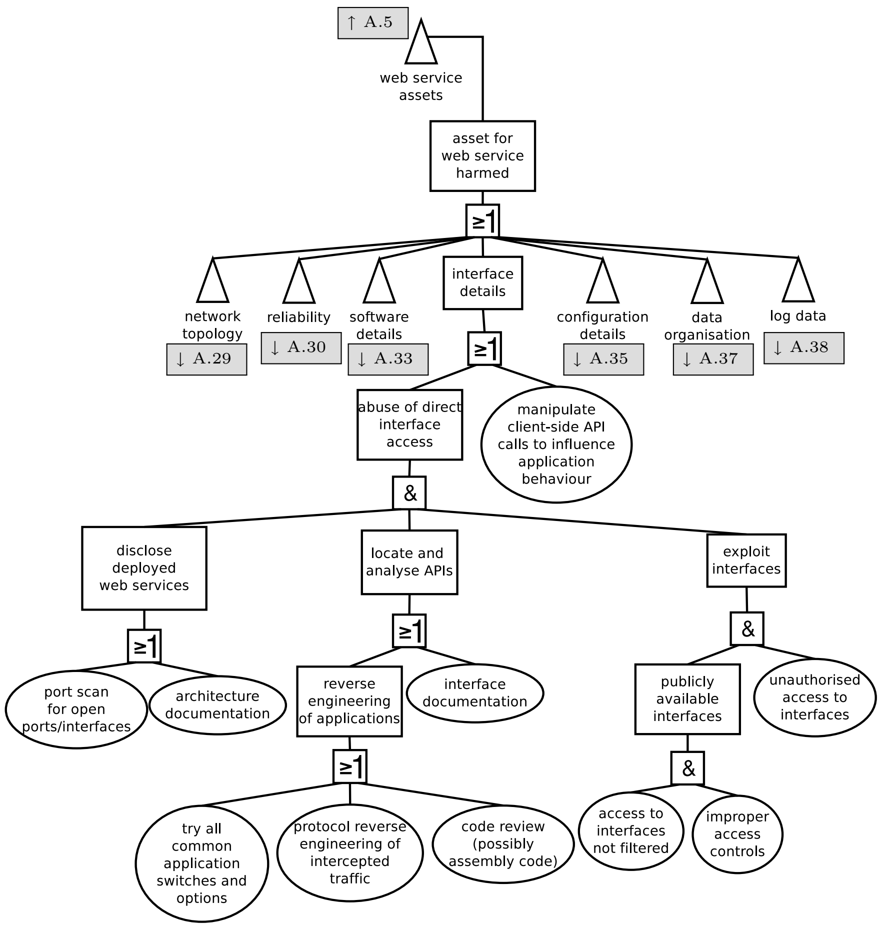

↑ A.5
web service assets
asset for web service harmed
≥1
network topology
↓ A.29
reliability
↓ A.30
software details
↓ A.33
interface details
configuration details
↓ A.35
data organisation
↓ A.37
log data
↓ A.38
≥1
abuse of direct interface access
manipulate client-side API calls to influence application behaviour
&
disclose deployed web services
locate and analyse APIs
exploit interfaces
≥1
port scan for open ports/interfaces
architecture documentation
≥1
reverse engineering of applications
interface documentation
&
publicly available interfaces
unauthorised access to interfaces
≥1
try all common application switches and options
protocol reverse engineering of intercepted traffic
code review (possibly assembly code)
&
access to interfaces not filtered
improper access controls

Figure A.28: Infrastructural basis

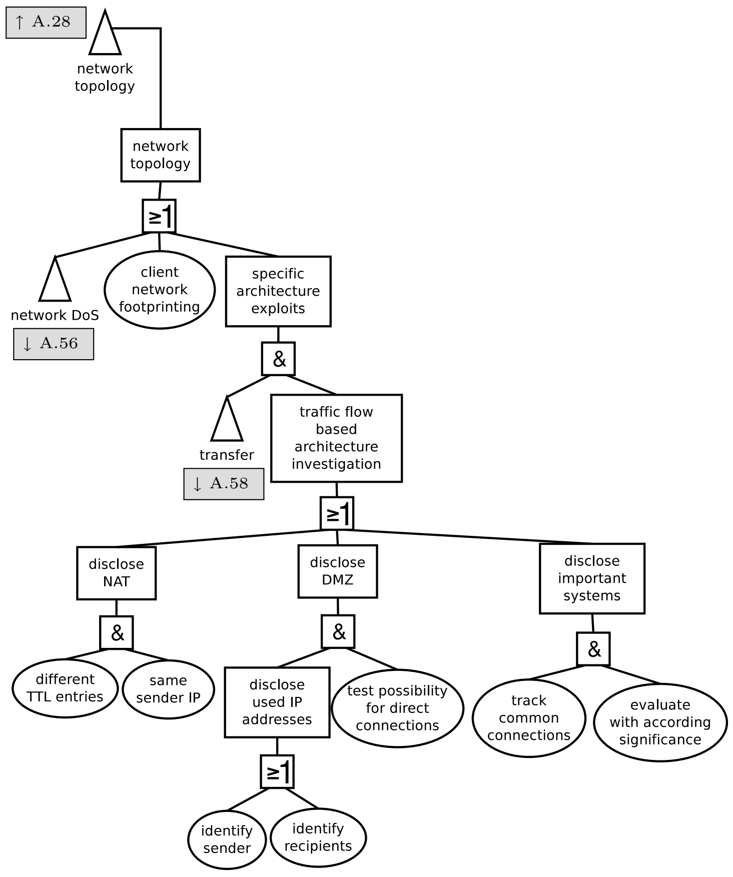

↑ A.28
network topology
network topology
≥1
network DoS
↓ A.56
client network footprinting
specific architecture exploits
&
transfer
↓ A.58
traffic flow based architecture investigation
≥1
disclose NAT
&
different TTL entries
same sender IP
disclose DMZ
&
disclose used IP addresses
≥1
identify sender
identify recipients
test possibility for direct connections
disclose important systems
&
track common connections
evaluate with according significance

Figure A.29: Network topology

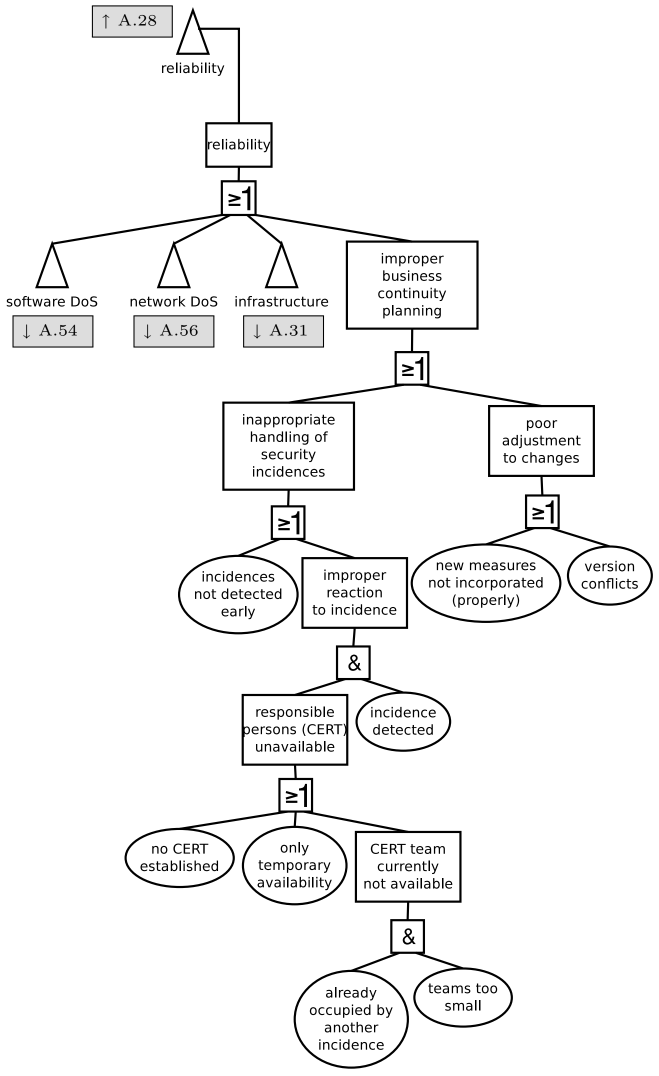

↑ A.28
reliability
reliability
≥1
software DoS
↓ A.54
network DoS
↓ A.56
infrastructure
↓ A.31
improper business continuity planning
≥1
inappropriate handling of security incidences
poor adjustment to changes
≥1
≥1
incidences not detected early
improper reaction to incidence
new measures not incorporated (properly)
version conflicts
&
responsible persons (CERT) unavailable
incidence detected
≥1
no CERT established
only temporary availability
CERT team currently not available
&
already occupied by another incidence
teams too small

Figure A.30: Reliability

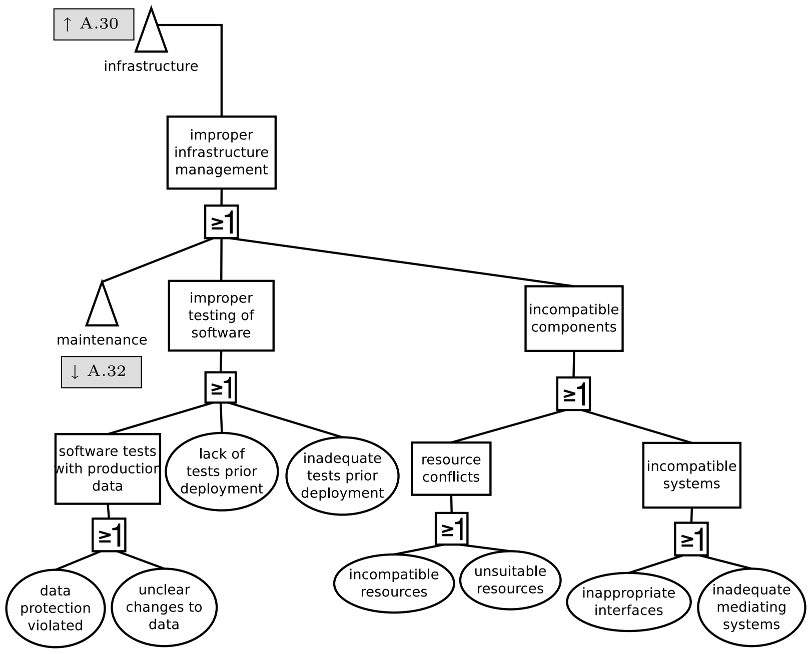


Figure A.31: Improper infrastructure management

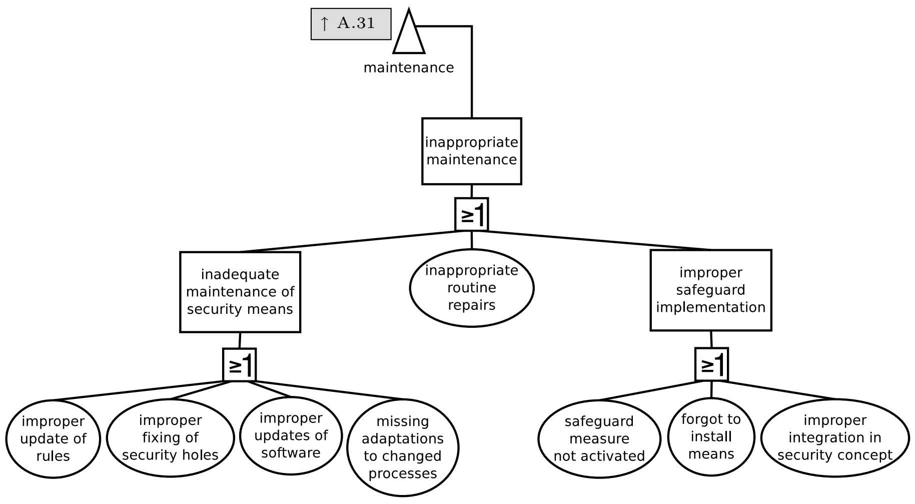

↑ A.31
maintenance
inappropriate maintenance
≥1
inadequate maintenance of security means
inappropriate routine repairs
improper safeguard implementation
≥1
improper update of rules
improper fixing of security holes
improper updates of software
missing adaptations to changed processes
≥1
safeguard measure not activated
forgot to install means
improper integration in security concept

Figure A.32: Inappropriate maintenance

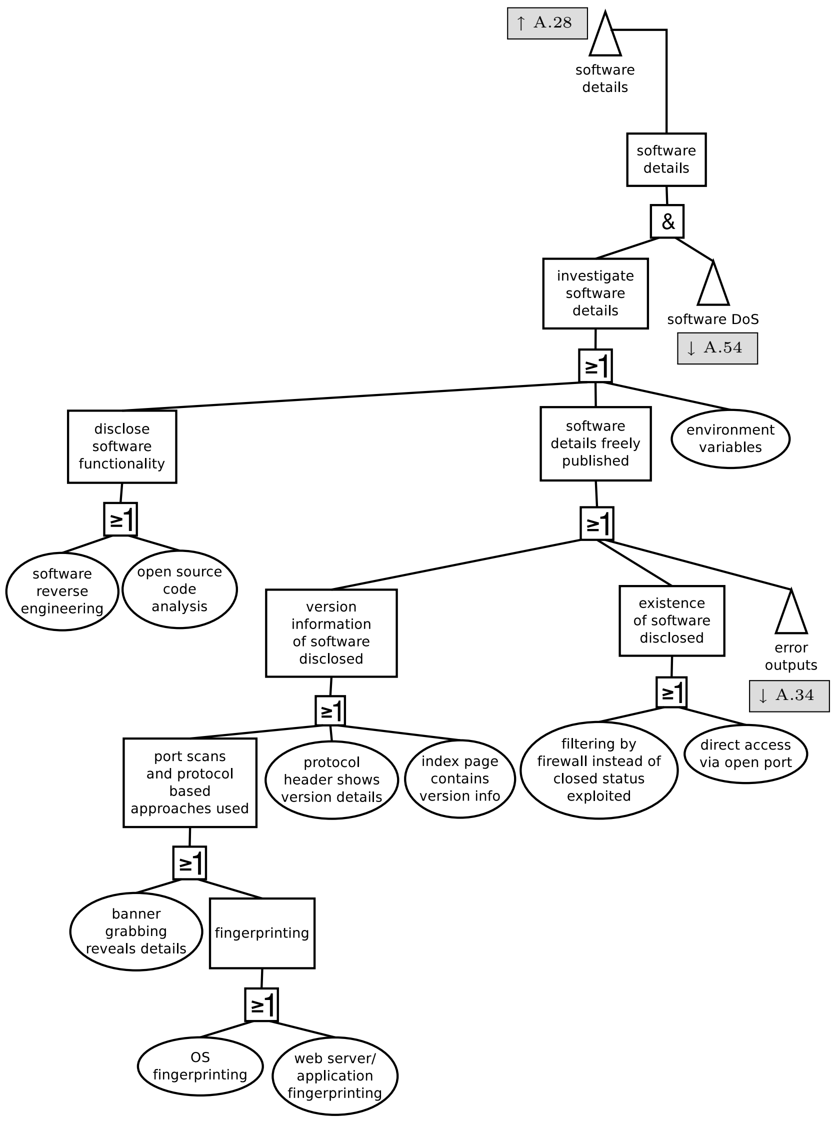

↑ A.28
software details
software details
&
investigate software details
software DoS
↓ A.54
≥1
disclose software functionality
software details freely published
environment variables
≥1
software reverse engineering
open source code analysis
≥1
version information of software disclosed
existence of software disclosed
error outputs
↓ A.34
≥1
port scans and protocol based approaches used
protocol header shows version details
index page contains version info
≥1
filtering by firewall instead of closed status exploited
direct access via open port
≥1
banner grabbing reveals details
fingerprinting
≥1
OS fingerprinting
web server/ application fingerprinting

Figure A.33: Software details

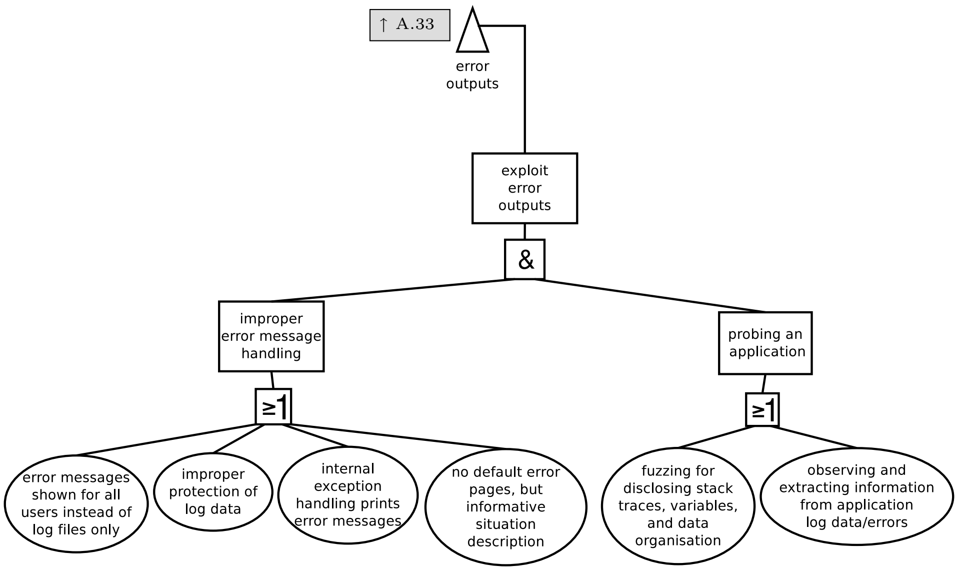

↑ A.33
error outputs
exploit error outputs
&
improper error message handling
probing an application
≥1
≥1
error messages shown for all users instead of log files only
improper protection of log data
internal exception handling prints error messages
no default error pages, but informative situation description
fuzzing for disclosing stack traces, variables, and data organisation
observing and extracting information from application log data/errors

Figure A.34: Exploit error output

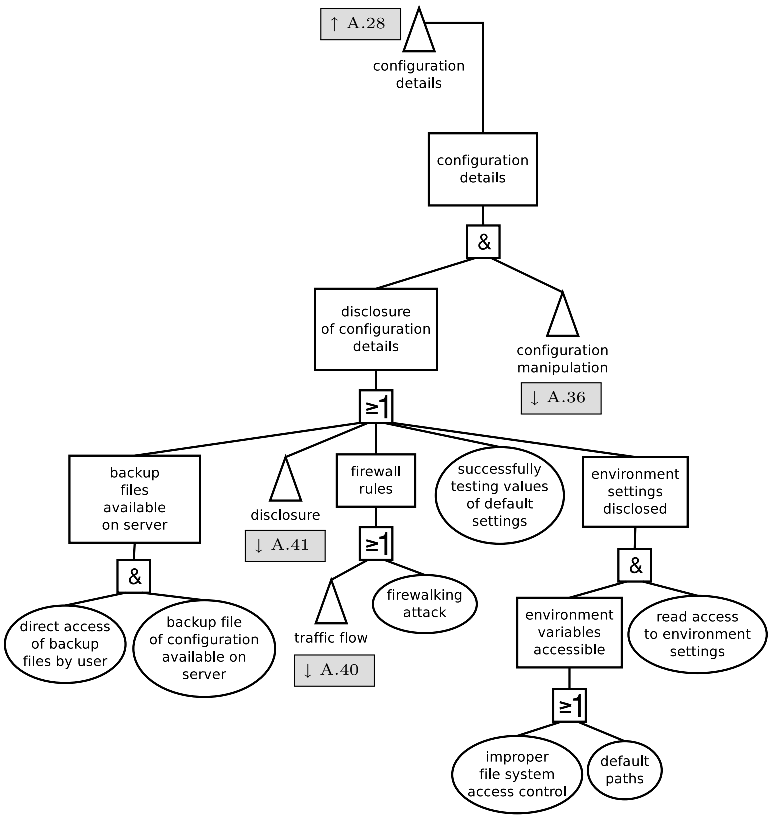

↑ A.28
configuration details
configuration details
&
disclosure of configuration details
configuration manipulation
↓ A.36
≥1
backup files available on server
disclosure
↓ A.41
firewall rules
successfully testing values of default settings
environment settings disclosed
&
direct access of backup files by user
backup file of configuration available on server
≥1
traffic flow
↓ A.40
firewalking attack
&
environment variables accessible
read access to environment settings
≥1
improper file system access control
default paths

Figure A.35: Configuration details

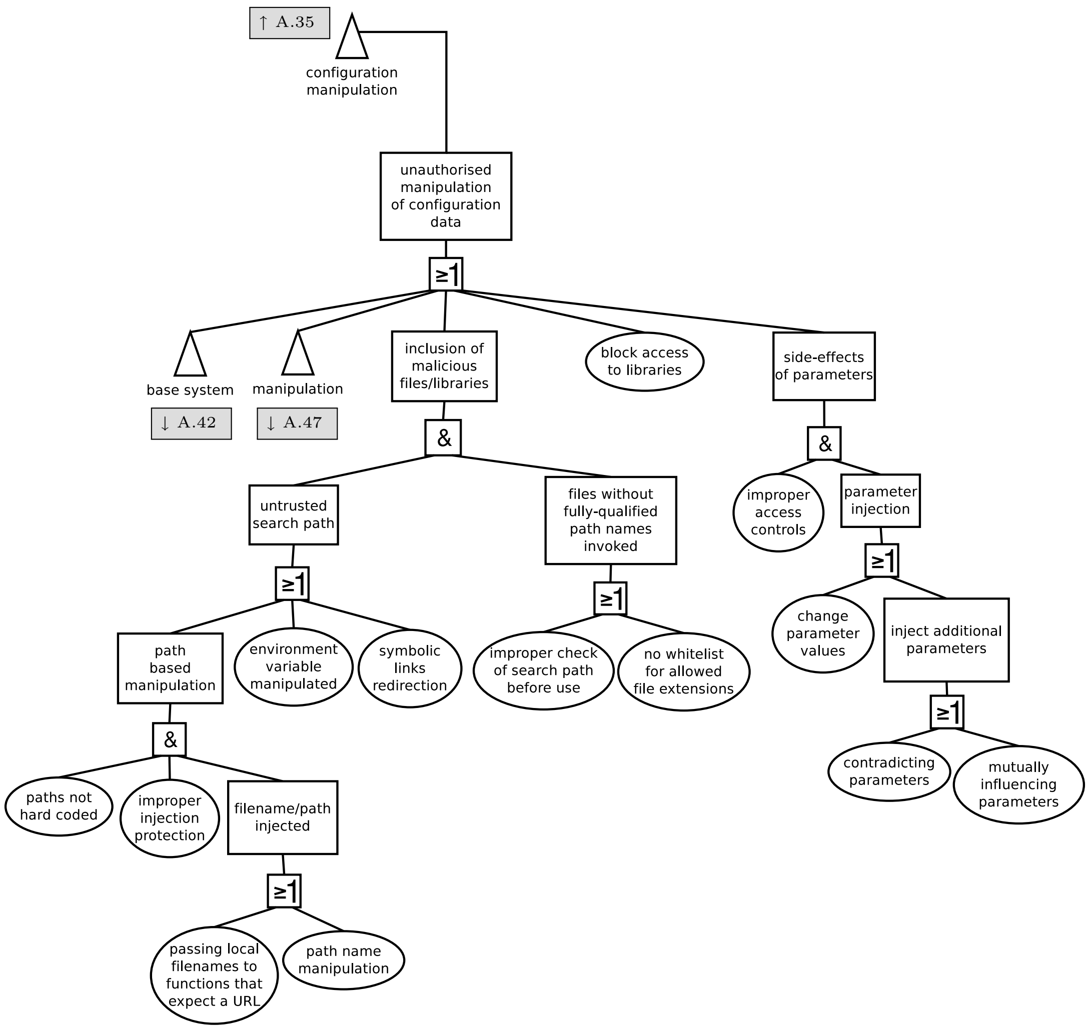

↑ A.35
configuration manipulation
unauthorised manipulation of configuration data
≥1
base system
↓ A.42
manipulation
↓ A.47
inclusion of malicious files/libraries
block access to libraries
side-effects of parameters
&
untrusted search path
files without fully-qualified path names invoked
&
improper access controls
parameter injection
≥1
path based manipulation
environment variable manipulated
symbolic links redirection
≥1
improper check of search path before use
no whitelist for allowed file extensions
≥1
change parameter values
inject additional parameters
≥1
contradicting parameters
mutually influencing parameters
&
paths not hard coded
improper injection protection
filename/path injected
≥1
passing local filenames to functions that expect a URL
path name manipulation

Figure A.36: Unauthorised manipulation of configuration data

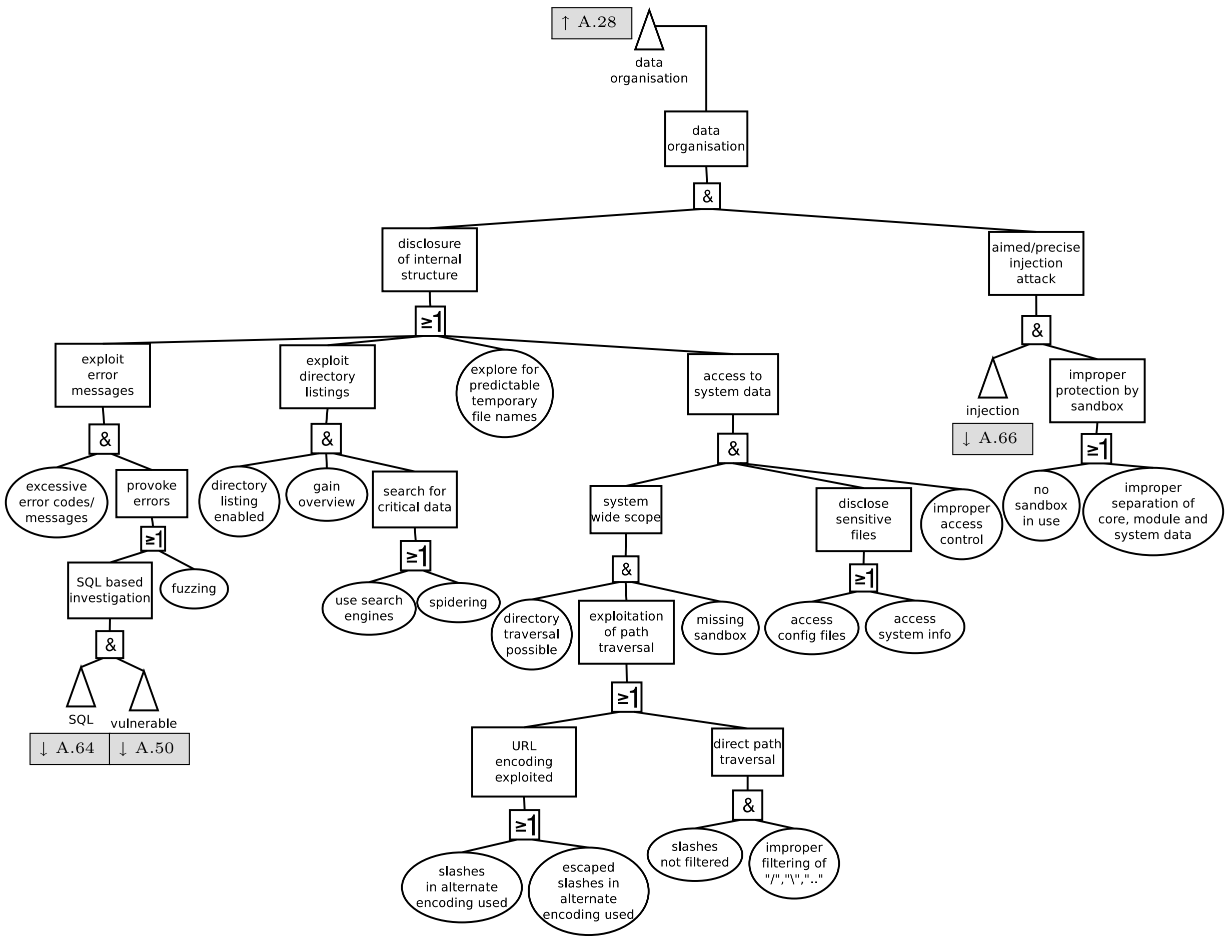


Figure A.37: Data organisation

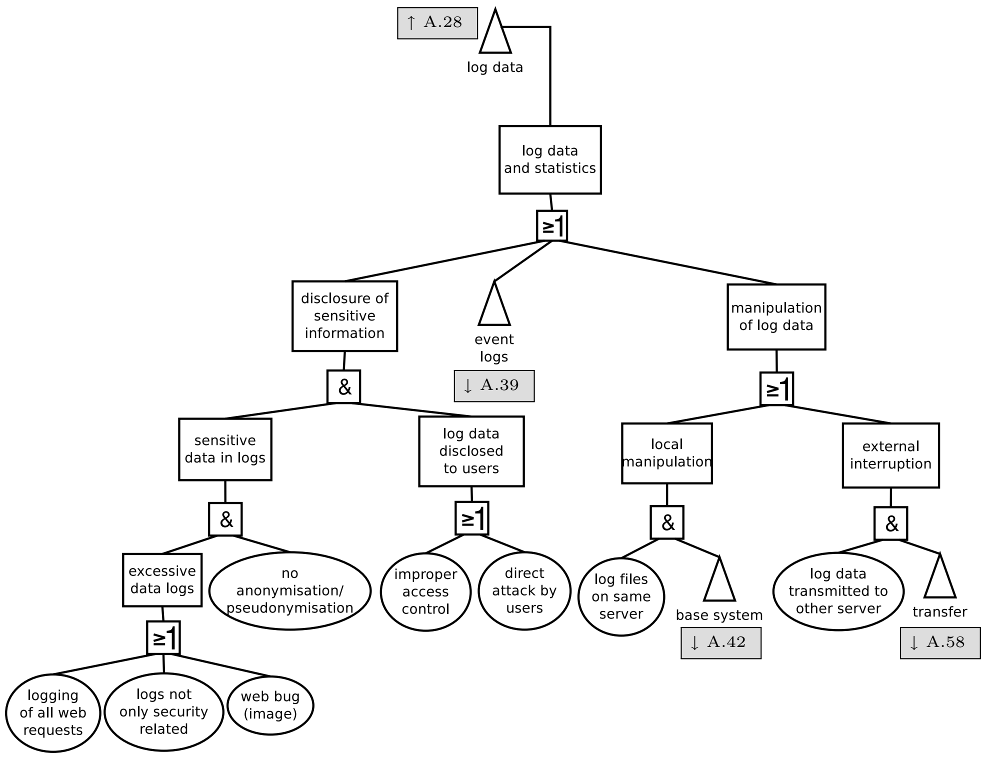


Figure A.38: Log data and statistics

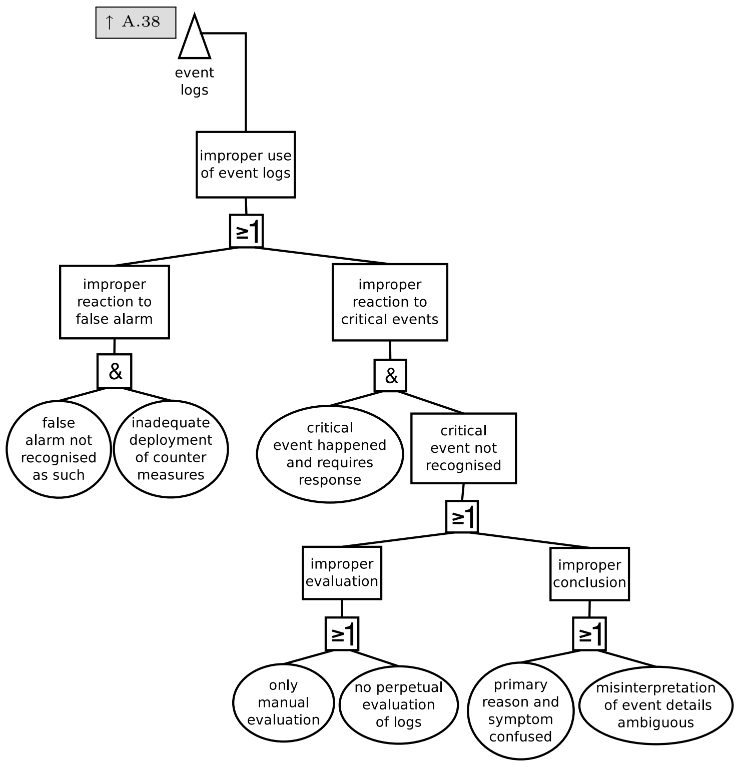

↑ A.38
event logs
improper use of event logs
≥1
improper reaction to false alarm
improper reaction to critical events
&
false alarm not recognised as such
inadequate deployment of counter measures
&
critical event happened and requires response
critical event not recognised
≥1
improper evaluation
improper conclusion
≥1
only manual evaluation
no perpetual evaluation of logs
≥1
primary reason and symptom confused
misinterpretation of event details ambiguous

Figure A.39: Improper use of event logs

A.3 Redundant Subtrees

Several subtrees of more generic relevance have to be included in redundant manner to appropriately state logic relations. In the following, seventeen redundant trees are presented that, again, may be broken into pieces for better fitting to the given page size. Note that for these redundant subtrees several references at the top-most transfer symbols indicate all figures that require this respective tree for inclusion. Figures that do only have a single reference at the top-most transfer symbol represent subelements of only one of these redundant tree.

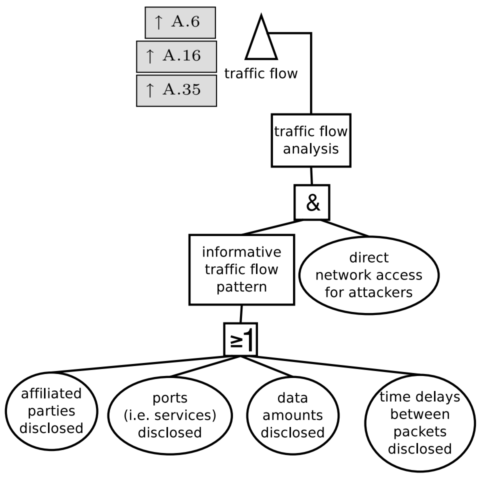

↑ A.6
↑ A.16
↑ A.35
traffic flow
traffic flow analysis
&
informative traffic flow pattern
direct network access for attackers
≥1
affiliated parties disclosed
ports (i.e. services) disclosed
data amounts disclosed
time delays between packets disclosed

Figure A.40: Traffic flow analysis

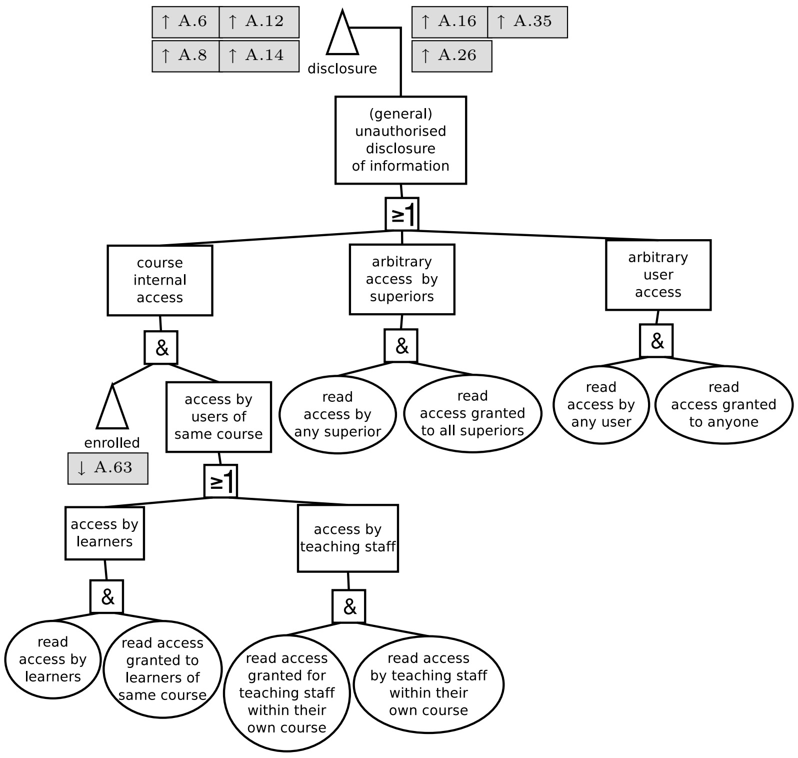

↑ A.6
↑ A.12
↑ A.8
↑ A.14
disclosure
↑ A.16
↑ A.35
↑ A.26
(general) unauthorised disclosure of information
≥1
course internal access
arbitrary access by superiors
arbitrary user access
&
enrolled
↓ A.63
access by users of same course
read access by any superior
read access granted to all superiors
read access by any user
read access granted to anyone
access by learners
access by teaching staff
read access by learners
read access granted to learners of same course
read access granted for teaching staff within their own course
read access by teaching staff within their own course

Figure A.41: (General) unauthorised disclosure of information

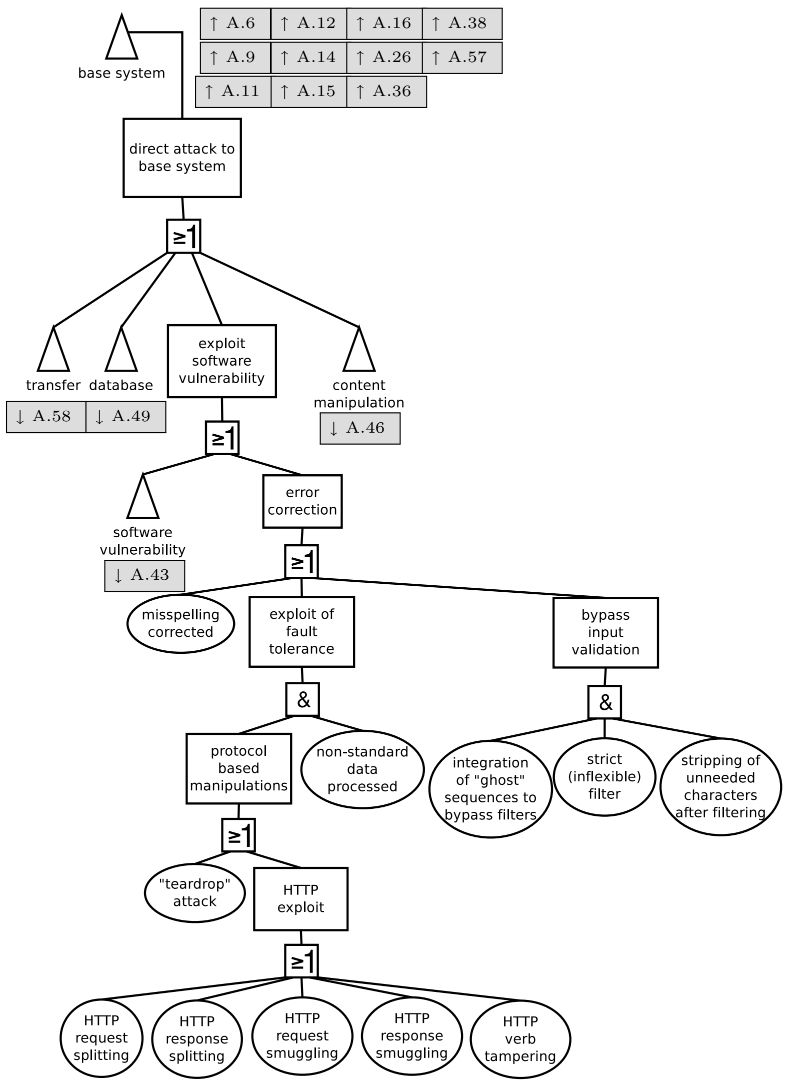


Figure A.42: Direct attack to base system

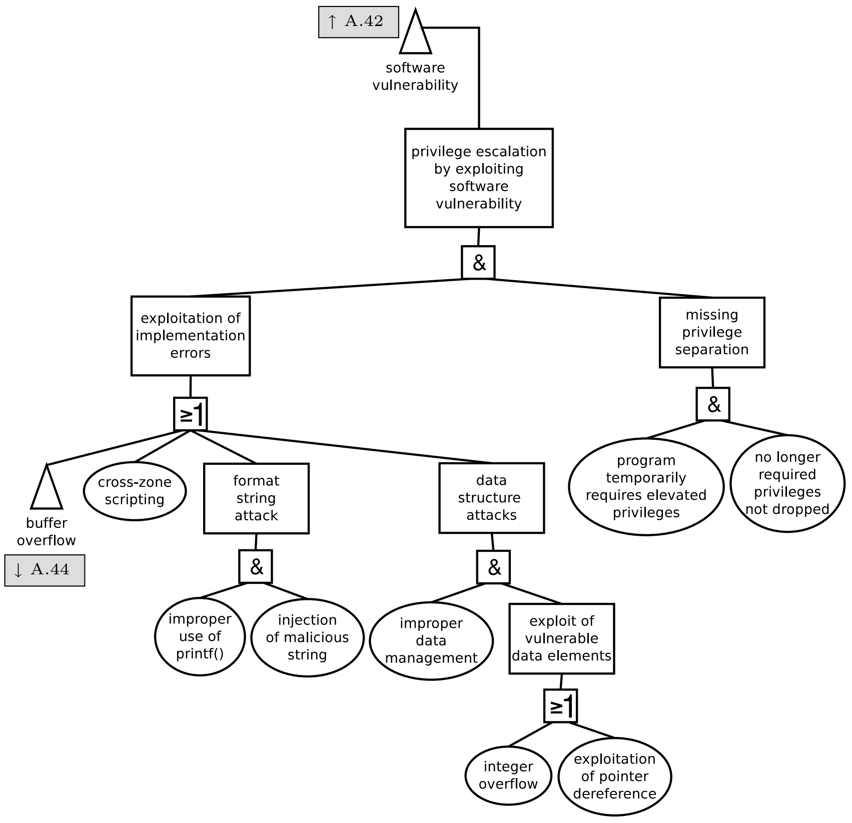

↑ A.42
software vulnerability
privilege escalation by exploiting software vulnerability
&
exploitation of implementation errors
missing privilege separation
≥1
&
buffer overflow
↓ A.44
cross-zone scripting
format string attack
data structure attacks
program temporarily requires elevated privileges
no longer required privileges not dropped
&
&
improper use of printf()
injection of malicious string
improper data management
exploit of vulnerable data elements
≥1
integer overflow
exploitation of pointer dereference

Figure A.43: Privilege escalation by exploiting software vulnerabilities

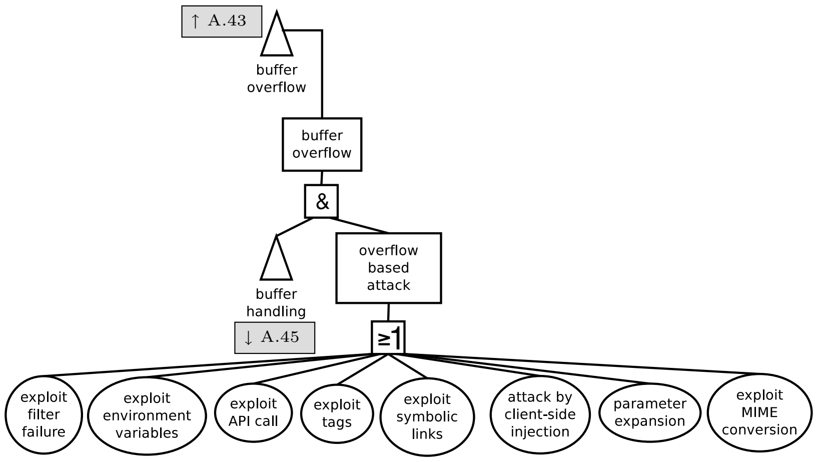


Figure A.44: Buffer overflow

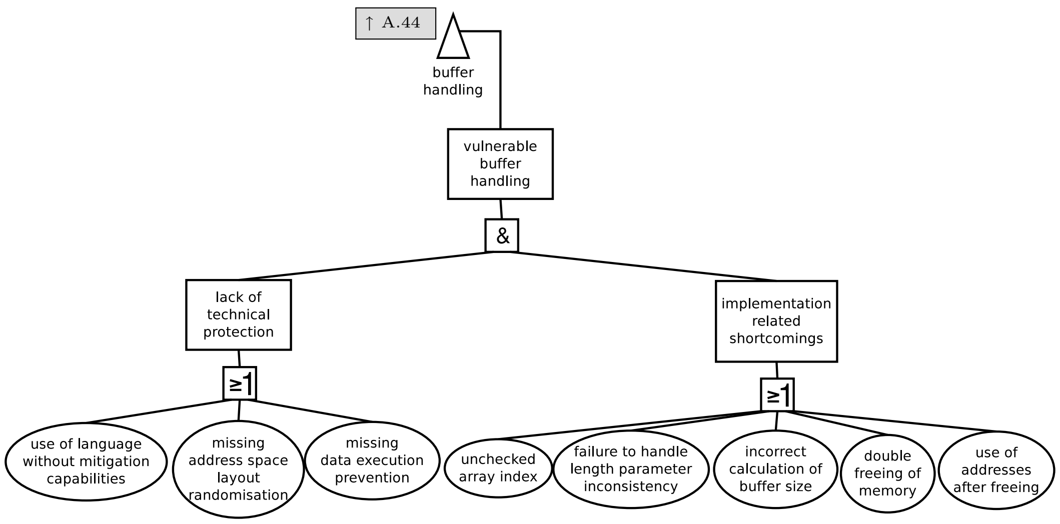

↑ A.44
buffer handling
vulnerable buffer handling
&
lack of technical protection
implementation related shortcomings
≥1
use of language without mitigation capabilities
missing address space layout randomisation
missing data execution prevention
≥1
unchecked array index
failure to handle length parameter inconsistency
incorrect calculation of buffer size
double freeing of memory
use of addresses after freeing

Figure A.45: Vulnerable buffer handling

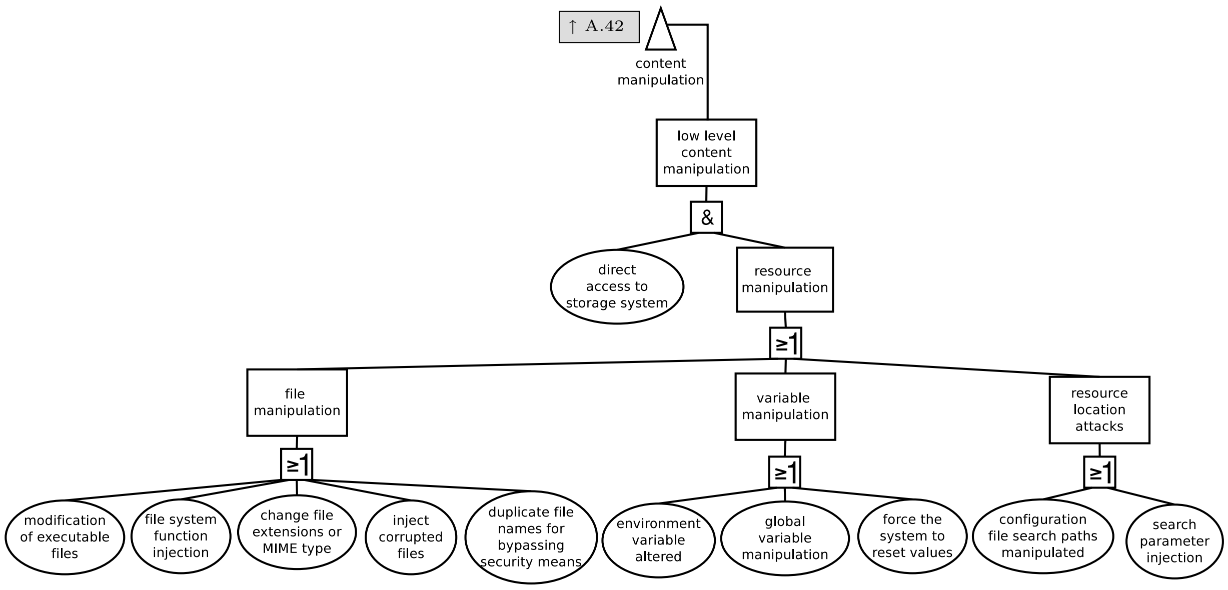

↑ A.42
content manipulation
low level content manipulation
&
direct access to storage system
resource manipulation
≥1
file manipulation
variable manipulation
resource location attacks
≥1
modification of executable files
file system function injection
change file extensions or MIME type
inject corrupted files
duplicate file names for bypassing security means
≥1
environment variable altered
global variable manipulation
force the system to reset values
≥1
configuration file search paths manipulated
search parameter injection

Figure A.46: Low level content manipulation

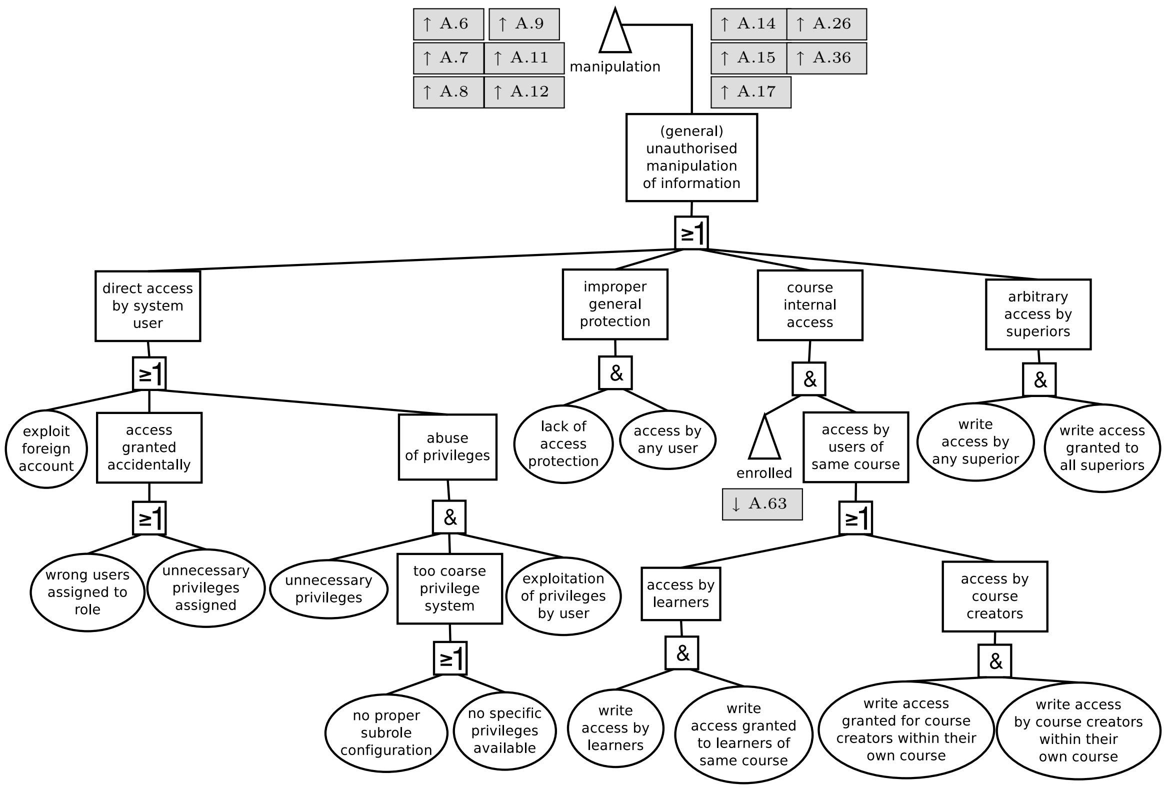

↑ A.6
↑ A.9
↑ A.7
↑ A.11
↑ A.8
↑ A.12
manipulation
↑ A.14
↑ A.26
↑ A.15
↑ A.36
↑ A.17
(general) unauthorised manipulation of information
≥1
direct access by system user
improper general protection
course internal access
arbitrary access by superiors
≥1
&
&
&
exploit foreign account
access granted accidentally
abuse of privileges
lack of access protection
access by any user
enrolled
↓ A.63
access by users of same course
write access by any superior
write access granted to all superiors
≥1
&
≥1
wrong users assigned to role
unnecessary privileges assigned
unnecessary privileges
too coarse privilege system
exploitation of privileges by user
access by learners
access by course creators
≥1
&
&
no proper subrole configuration
no specific privileges available
write access by learners
write access granted to learners of same course
write access granted for course creators within their own course
write access by course creators within their own course

Figure A.47: (General) unauthorised manipulation of information

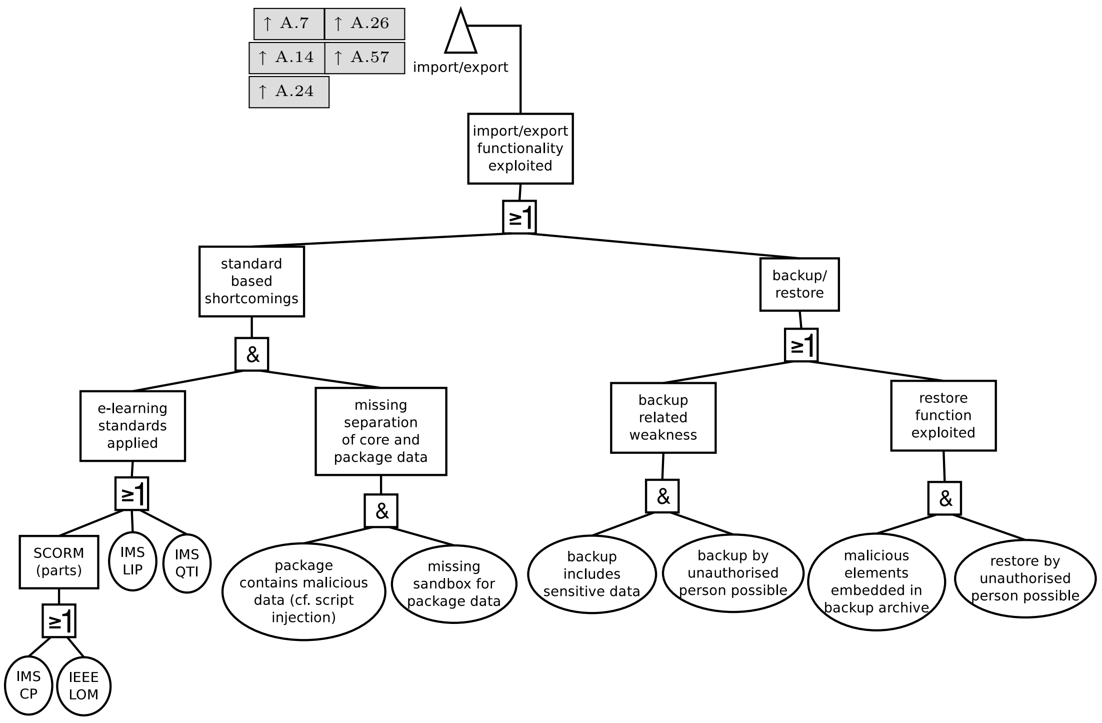

↑ A.7
↑ A.26
↑ A.14
↑ A.57
↑ A.24
import/export
import/export functionality exploited
≥1
standard based shortcomings
backup/ restore
&
≥1
e-learning standards applied
missing separation of core and package data
backup related weakness
restore function exploited
≥1
&
&
&
SCORM (parts)
IMS LIP
IMS QTI
package contains malicious data (cf. script injection)
missing sandbox for package data
backup includes sensitive data
backup by unauthorised person possible
malicious elements embedded in backup archive
restore by unauthorised person possible
≥1
IMS CP
IEEE LOM

Figure A.48: Import/export functionality exploited

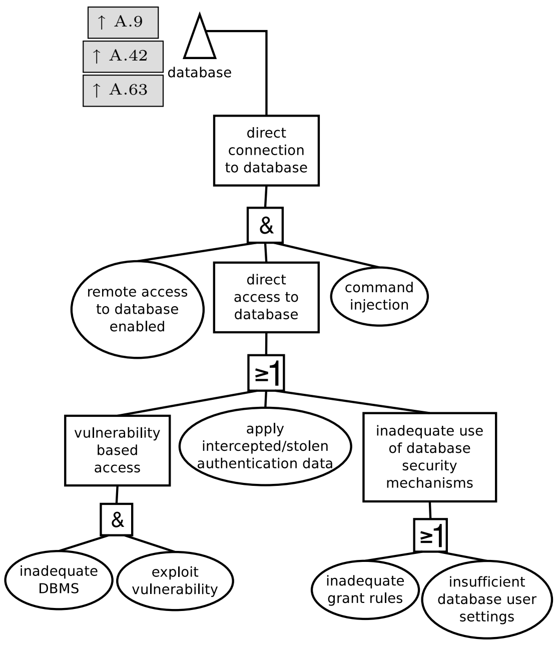

↑ A.9
↑ A.42
↑ A.63
database
direct connection to database
&
remote access to database enabled
direct access to database
command injection
≥1
vulnerability based access
apply intercepted/stolen authentication data
inadequate use of database security mechanisms
&
inadequate DBMS
exploit vulnerability
≥1
inadequate grant rules
insufficient database user settings

Figure A.49: Direct connection to database

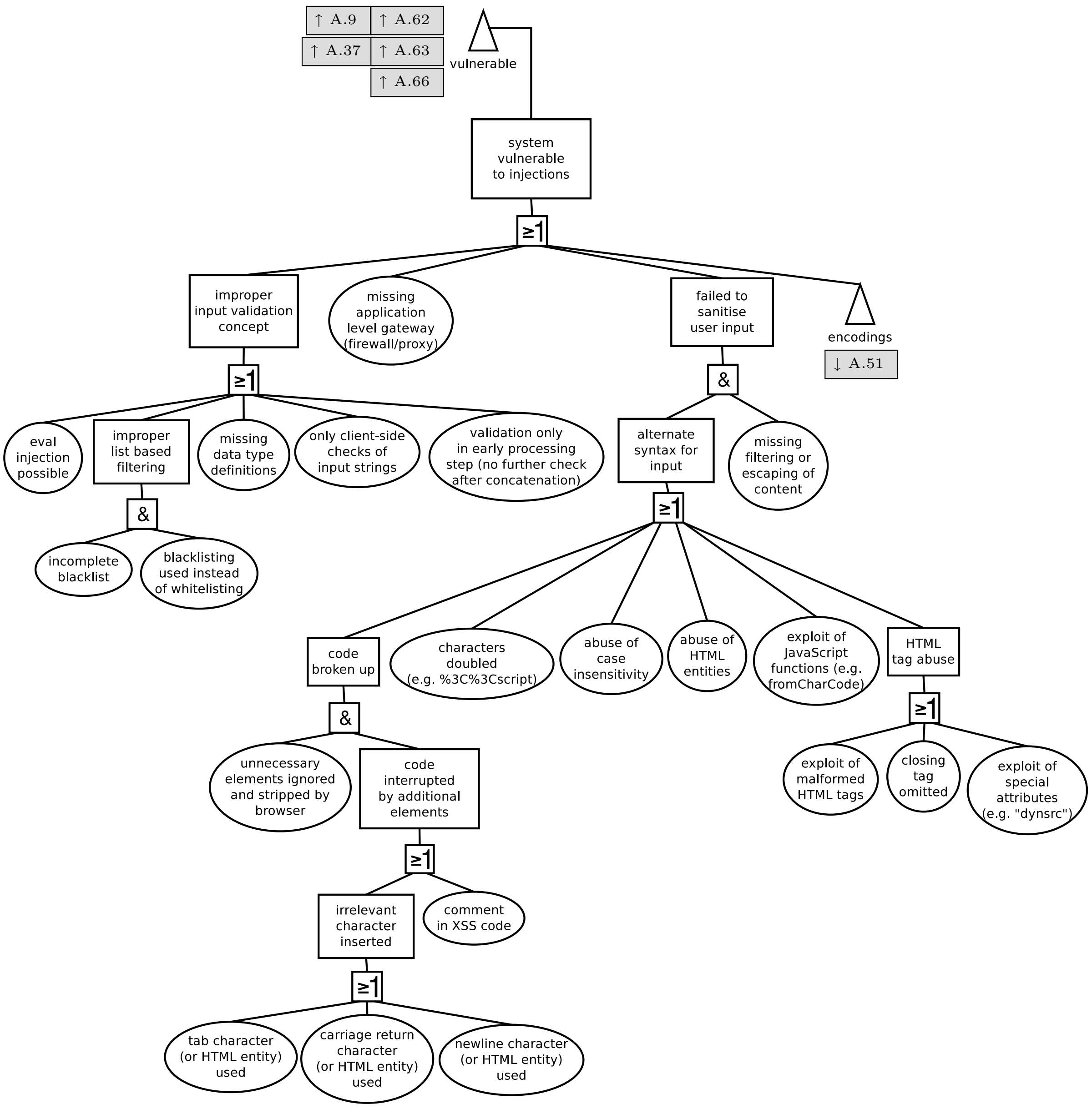

↑ A.9
↑ A.62
↑ A.37
↑ A.63
↑ A.66
vulnerable
system vulnerable to injections
≥1
improper input validation concept
missing application level gateway (firewall/proxy)
failed to sanitise user input
encodings
↓ A.51
≥1
eval injection possible
improper list based filtering
missing data type definitions
only client-side checks of input strings
validation only in early processing step (no further check after concatenation)
&
incomplete blacklist
blacklisting used instead of whitelisting
&
alternate syntax for input
missing filtering or escaping of content
≥1
code broken up
characters doubled (e.g. %3C%3Cscript)
abuse of case insensitivity
abuse of HTML entities
exploit of JavaScript functions (e.g. fromCharCode)
HTML tag abuse
&
unnecessary elements ignored and stripped by browser
code interrupted by additional elements
≥1
irrelevant character inserted
comment in XSS code
≥1
tab character (or HTML entity) used
carriage return character (or HTML entity) used
newline character (or HTML entity) used
≥1
exploit of malformed HTML tags
closing tag omitted
exploit of special attributes (e.g. "dynsrc")

Figure A.50: System vulnerable to injections

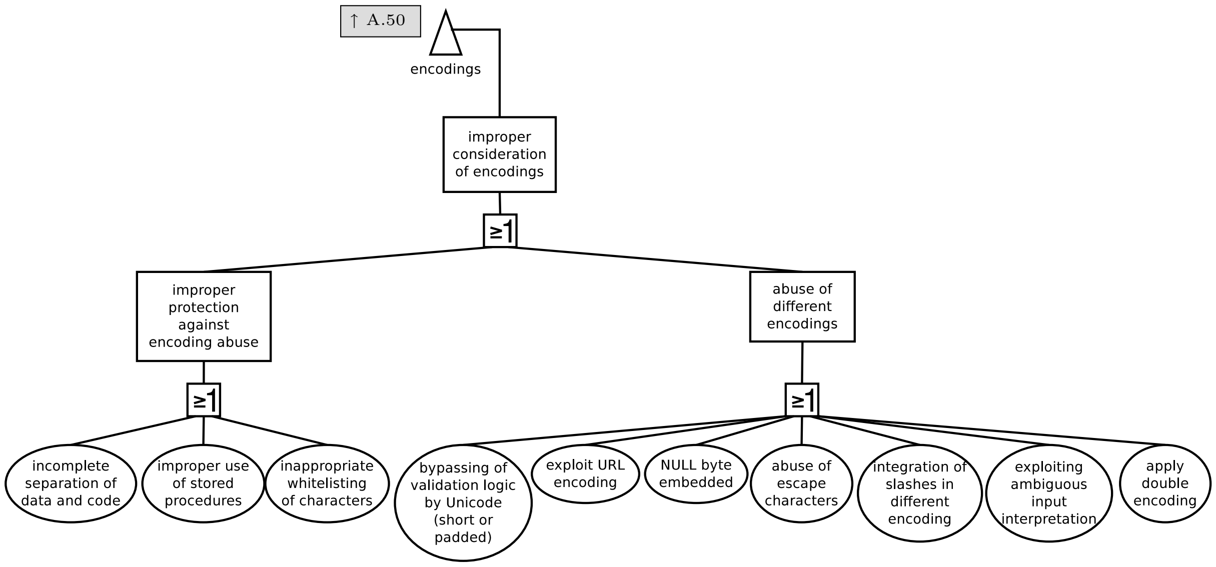

↑ A.50
encodings
improper consideration of encodings
≥1
improper protection against encoding abuse
abuse of different encodings
≥1
incomplete separation of data and code
improper use of stored procedures
inappropriate whitelisting of characters
≥1
bypassing of validation logic by Unicode (short or padded)
exploit URL encoding
NULL byte embedded
abuse of escape characters
integration of slashes in different encoding
exploiting ambiguous input interpretation
apply double encoding

Figure A.51: Improper consideration of encodings

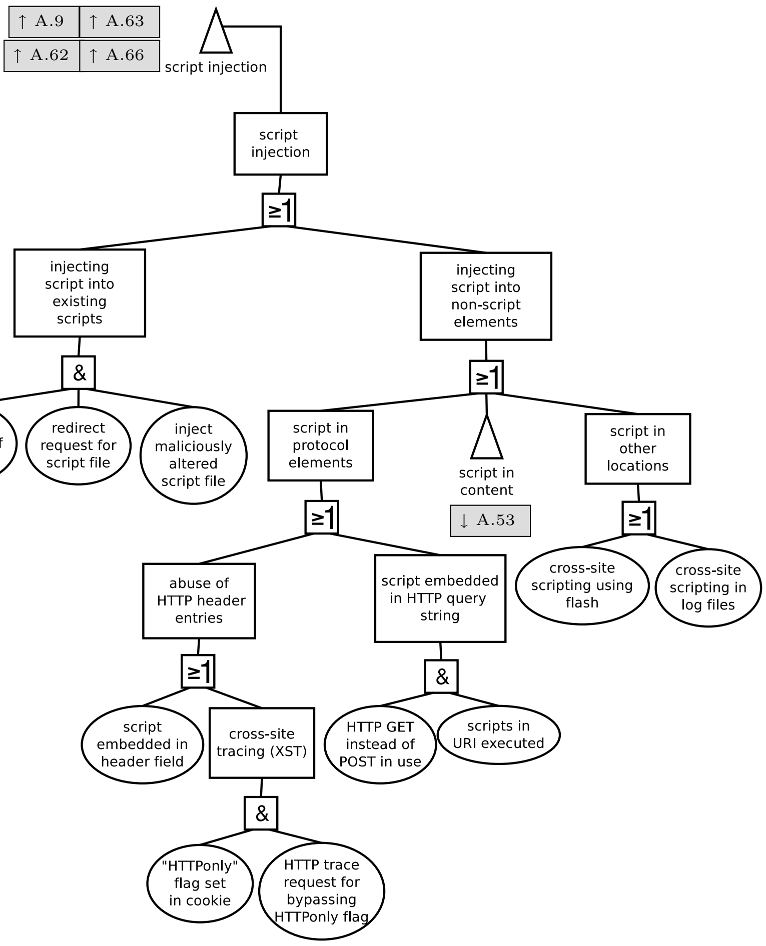

↑ A.9
↑ A.63
↑ A.62
↑ A.66
script injection
script injection
≥1
injecting script into existing scripts
injecting script into non-script elements
&
external storage of scripts
redirect request for script file
inject maliciously altered script file
≥1
script in protocol elements
script in content
↓ A.53
script in other locations
≥1
abuse of HTTP header entries
script embedded in HTTP query string
≥1
cross-site scripting using flash
cross-site scripting in log files
≥1
script embedded in header field
cross-site tracing (XST)
&
HTTP GET instead of POST in use
scripts in URI executed
&
"HTTPonly" flag set in cookie
HTTP trace request for bypassing HTTPonly flag

Figure A.52: Script injection

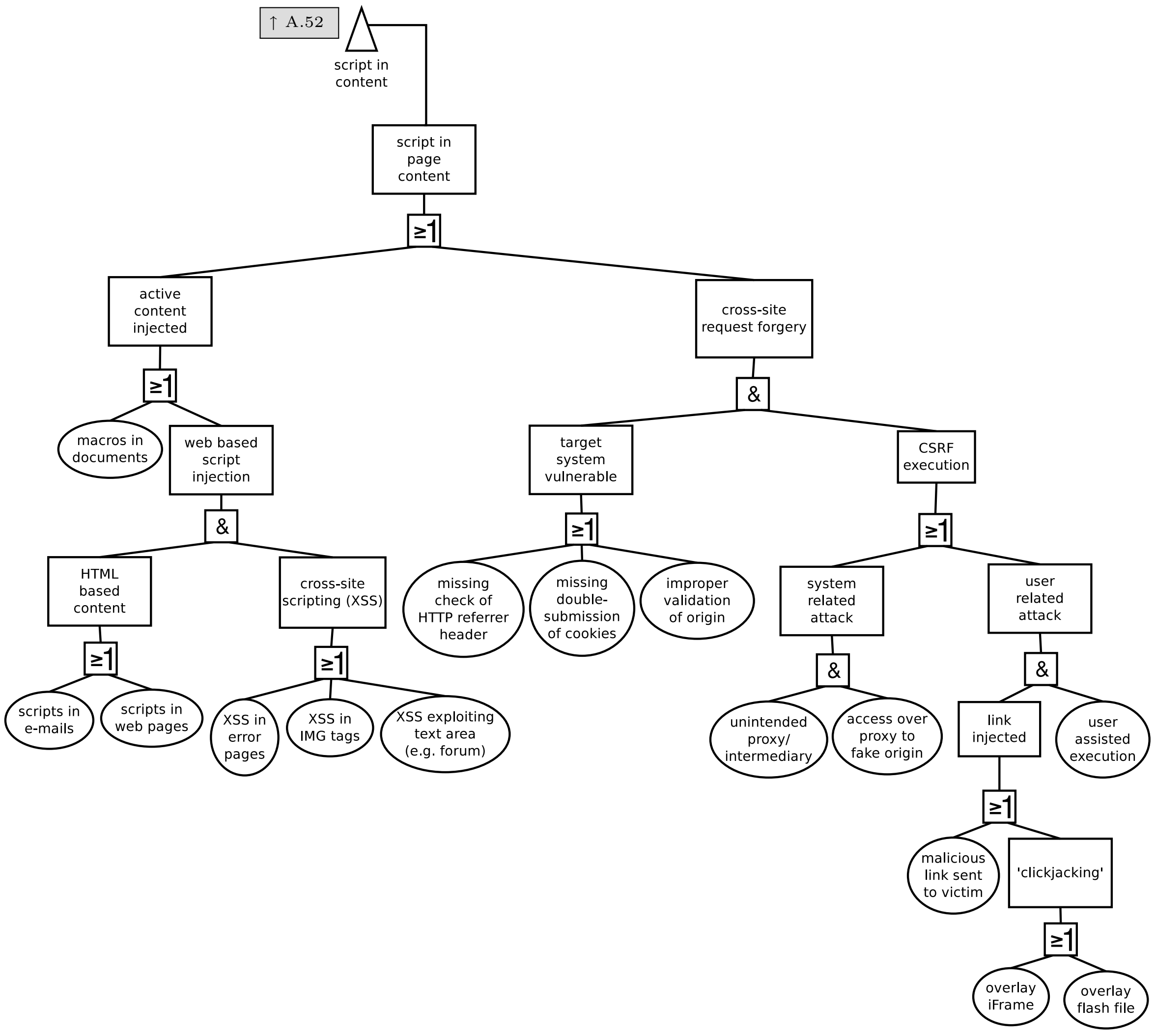


Figure A.53: Script in page content

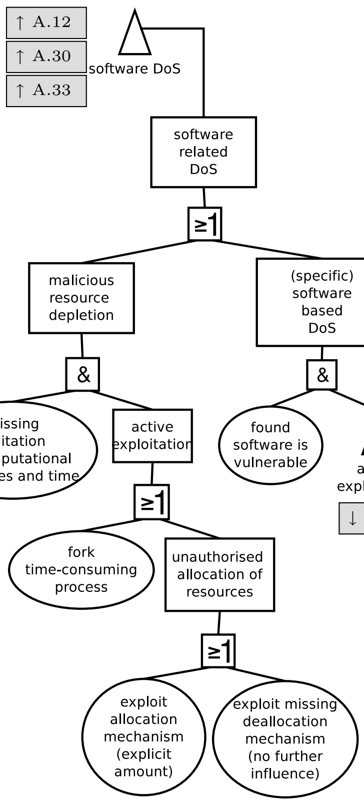

↑ A.12
↑ A.30
↑ A.33
software DoS
software related DoS
≥1
malicious resource depletion
(specific) software based DoS
&
&
missing limitation of computational resources and time
active exploitation
found software is vulnerable
active exploitation
↓ A.55
≥1
fork time-consuming process
unauthorised allocation of resources
≥1
exploit allocation mechanism (explicit amount)
exploit missing deallocation mechanism (no further influence)

Figure A.54: Software related denial of service (DoS)

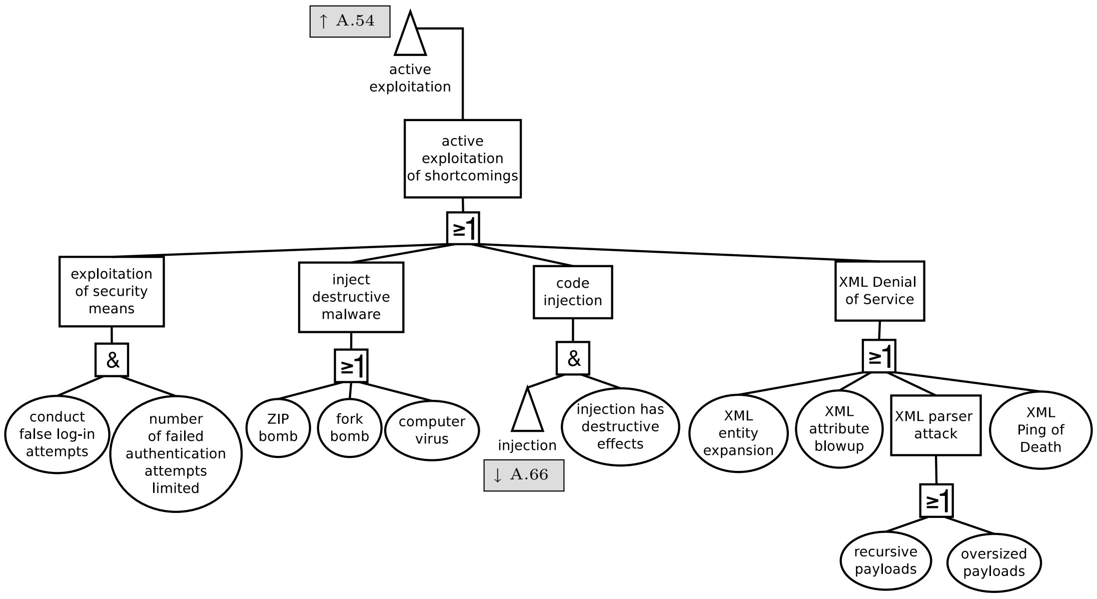


Figure A.55: Active exploitation of shortcomings

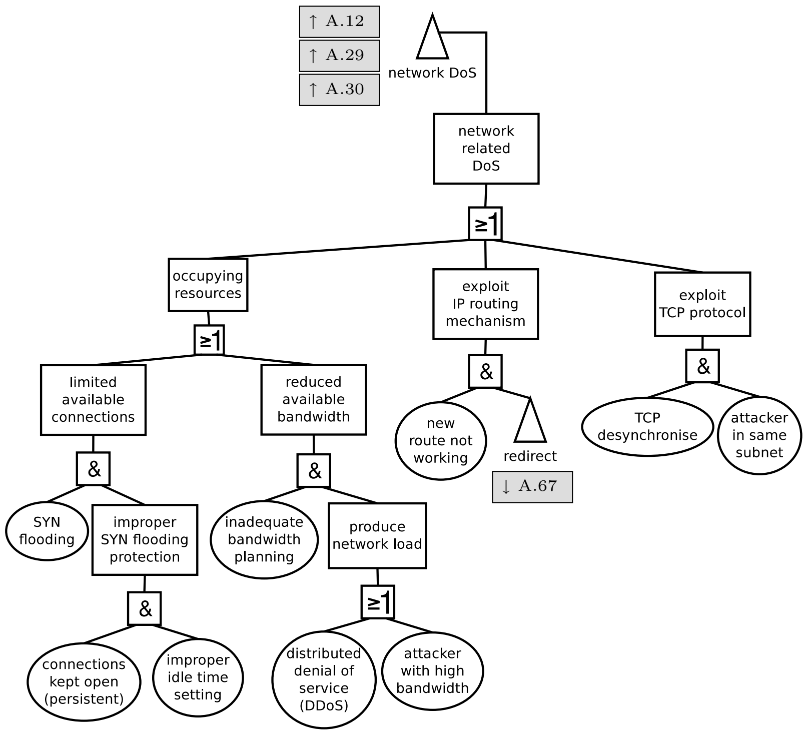

↑ A.12
↑ A.29
↑ A.30
network DoS
network related DoS
≥1
occupying resources
exploit IP routing mechanism
exploit TCP protocol
≥1
limited available connections
reduced available bandwidth
&
new route not working
redirect
↓ A.67
&
TCP desynchronise
attacker in same subnet
&
SYN flooding
improper SYN flooding protection
&
inadequate bandwidth planning
produce network load
&
connections kept open (persistent)
improper idle time setting
≥1
distributed denial of service (DDoS)
attacker with high bandwidth

Figure A.56: Network related denial of service (DoS)

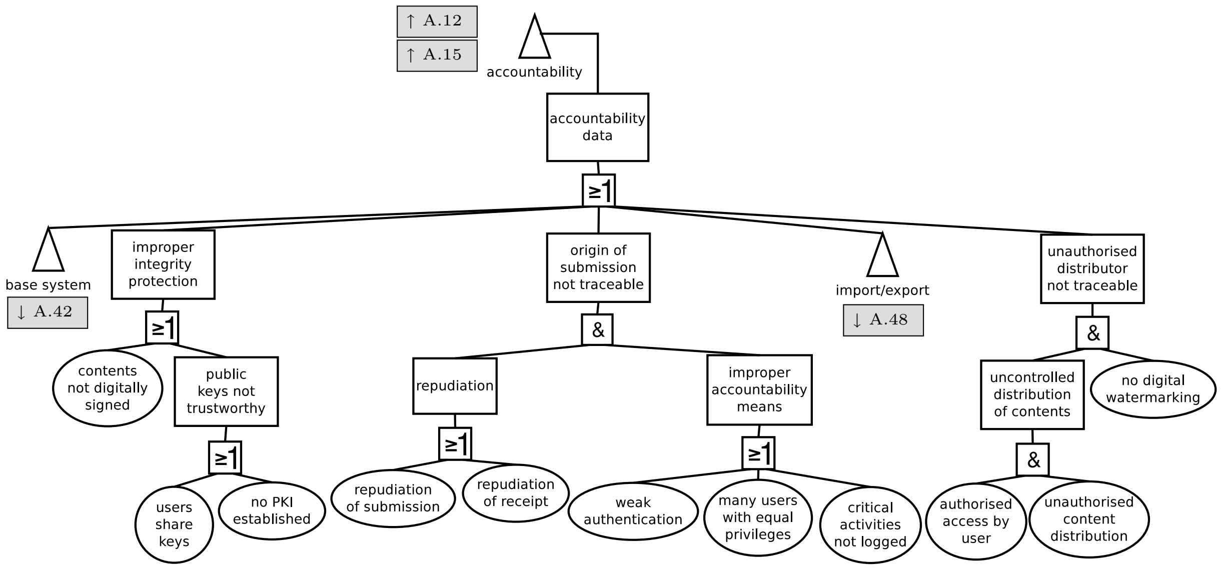

↑ A.12
↑ A.15
accountability
accountability data
≥1
base system
↓ A.42
improper integrity protection
≥1
contents not digitally signed
public keys not trustworthy
≥1
users share keys
no PKI established
origin of submission not traceable
&
repudiation
≥1
repudiation of submission
repudiation of receipt
improper accountability means
≥1
weak authentication
many users with equal privileges
critical activities not logged
import/export
↓ A.48
unauthorised distributor not traceable
&
uncontrolled distribution of contents
&
authorised access by user
unauthorised content distribution
no digital watermarking

Figure A.57: Accountability data

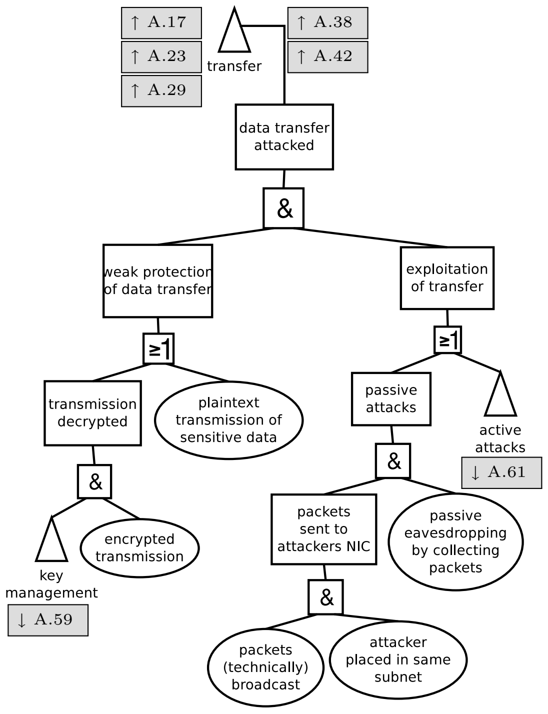

↑ A.17
↑ A.23
↑ A.29
transfer
↑ A.38
↑ A.42
data transfer attacked
&
weak protection of data transfer
≥1
transmission decrypted
&
key management
↓ A.59
encrypted transmission
plaintext transmission of sensitive data
exploitation of transfer
≥1
passive attacks
&
packets sent to attackers NIC
&
packets (technically) broadcast
attacker placed in same subnet
passive eavesdropping by collecting packets
active attacks
↓ A.61

Figure A.58: Data transfer attacked

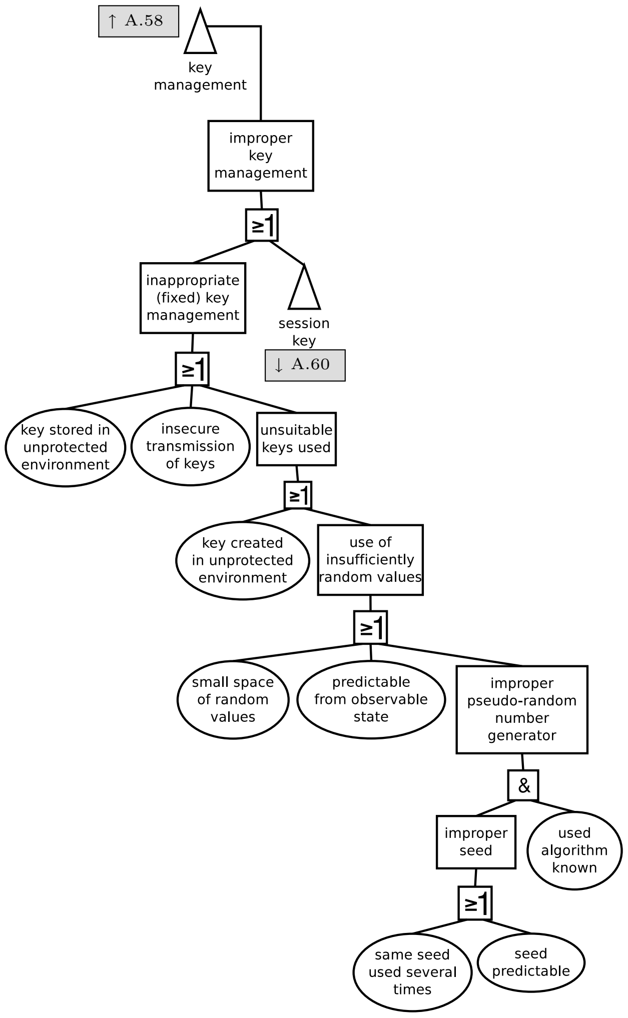

↑ A.58
key management
improper key management
≥1
inappropriate (fixed) key management
session key
↓ A.60
≥1
key stored in unprotected environment
insecure transmission of keys
unsuitable keys used
≥1
key created in unprotected environment
use of insufficiently random values
≥1
small space of random values
predictable from observable state
improper pseudo-random number generator
&
improper seed
used algorithm known
≥1
same seed used several times
seed predictable

Figure A.59: Improper key management

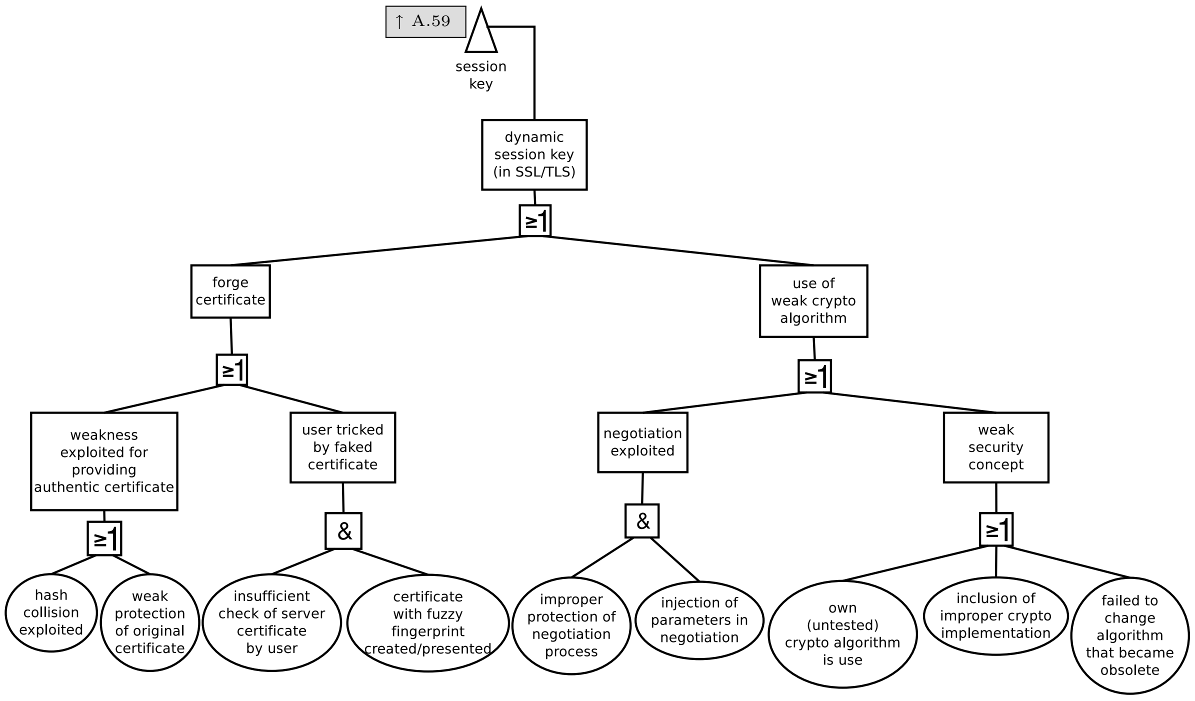

↑ A.59
session key
dynamic session key (in SSL/TLS)
≥1
forge certificate
use of weak crypto algorithm
≥1
weakness exploited for providing authentic certificate
user tricked by faked certificate
≥1
negotiation exploited
weak security concept
≥1
hash collision exploited
weak protection of original certificate
&
insufficient check of server certificate by user
certificate with fuzzy fingerprint created/presented
&
improper protection of negotiation process
injection of parameters in negotiation
≥1
own (untested) crypto algorithm is use
inclusion of improper crypto implementation
failed to change algorithm that became obsolete

Figure A.60: Dynamic session key

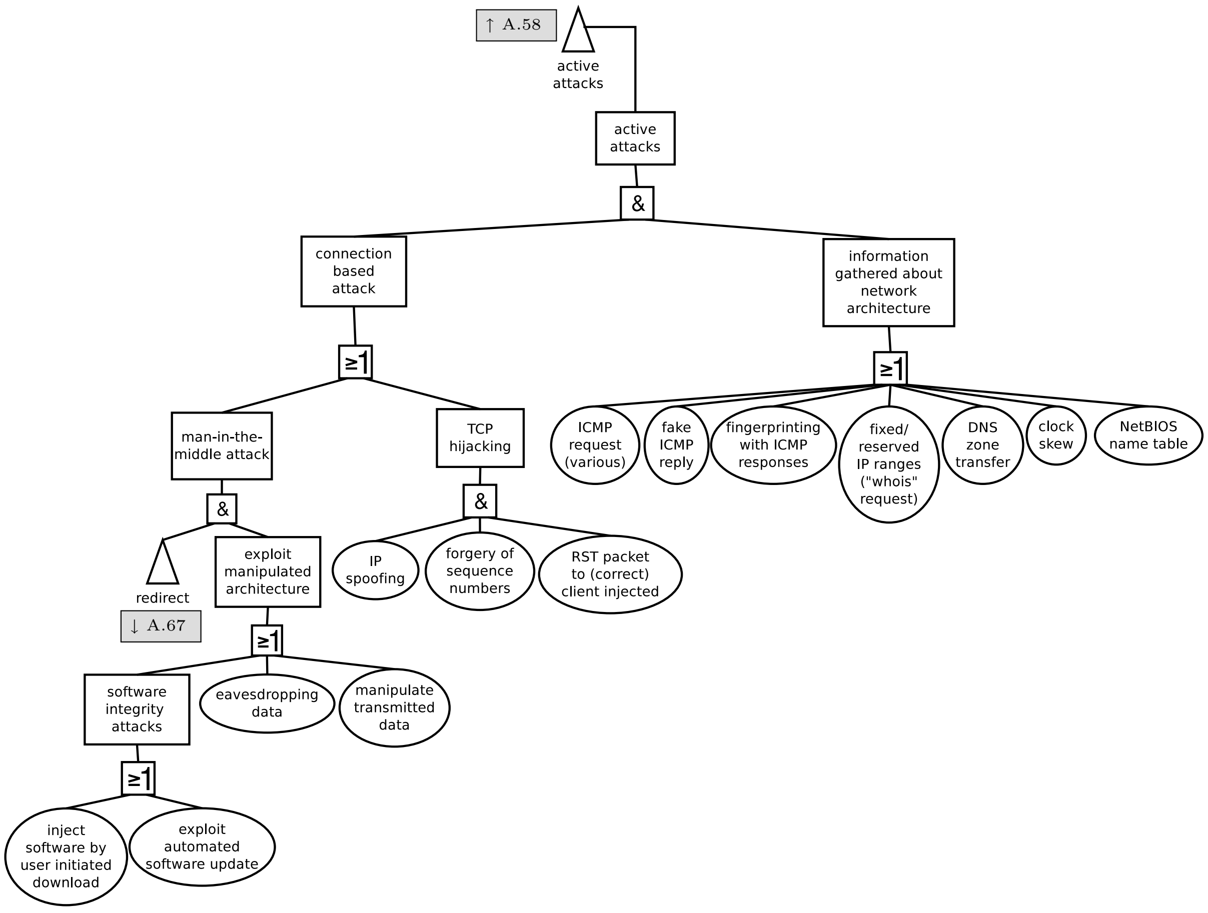

↑ A.58
active attacks
active attacks
&
connection based attack
information gathered about network architecture
≥1
≥1
man-in-the-middle attack
TCP hijacking
ICMP request (various)
fake ICMP reply
fingerprinting with ICMP responses
fixed/ reserved IP ranges ("whois" request)
DNS zone transfer
clock skew
NetBIOS name table
&
&
redirect
↓ A.67
exploit manipulated architecture
IP spoofing
forgery of sequence numbers
RST packet to (correct) client injected
≥1
software integrity attacks
eavesdropping data
manipulate transmitted data
≥1
inject software by user initiated download
exploit automated software update

Figure A.61: Active attacks

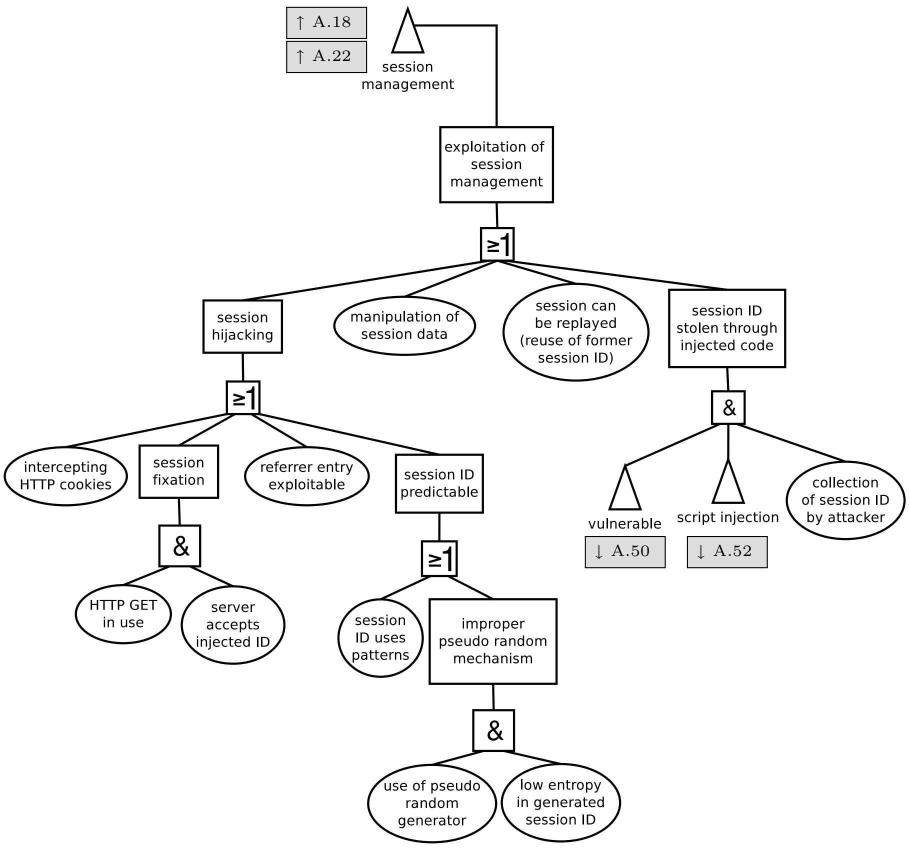

↑ A.18
↑ A.22
session management
exploitation of session management
≥1
session hijacking
manipulation of session data
session can be replayed (reuse of former session ID)
session ID stolen through injected code
≥1
intercepting HTTP cookies
session fixation
referrer entry exploitable
session ID predictable
&
HTTP GET in use
server accepts injected ID
≥1
session ID uses patterns
improper pseudo random mechanism
&
use of pseudo random generator
low entropy in generated session ID
&
vulnerable
↓ A.50
script injection
↓ A.52
collection of session ID by attacker

Figure A.62: Exploitation of session management

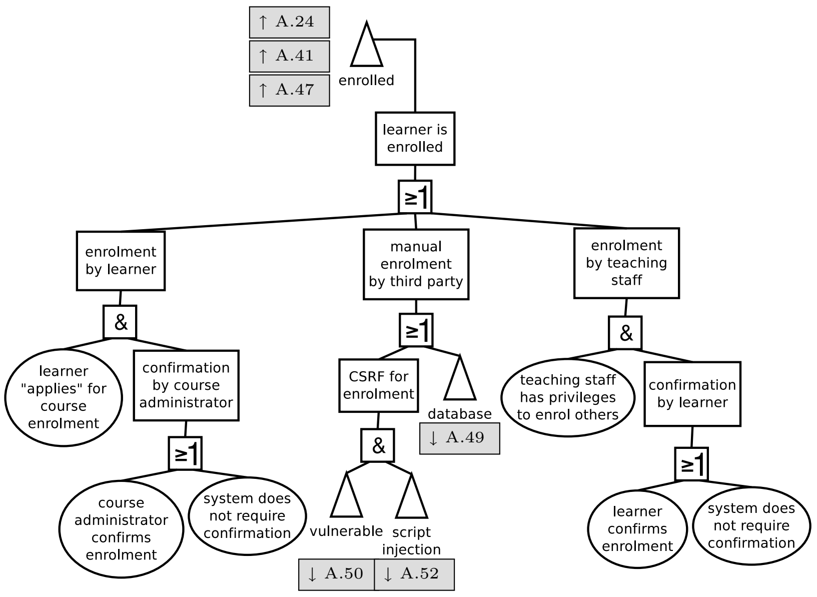

↑ A.24
↑ A.41
↑ A.47
enrolled
learner is enrolled
≥1
enrolment by learner
manual enrolment by third party
enrolment by teaching staff
&
learner "applies" for course enrolment
confirmation by course administrator
≥1
course administrator confirms enrolment
system does not require confirmation
≥1
CSRF for enrolment
database
↓ A.49
&
vulnerable
script injection
↓ A.50
↓ A.52
&
teaching staff has privileges to enrol others
confirmation by learner
≥1
learner confirms enrolment
system does not require confirmation

Figure A.63: Learner is enrolled

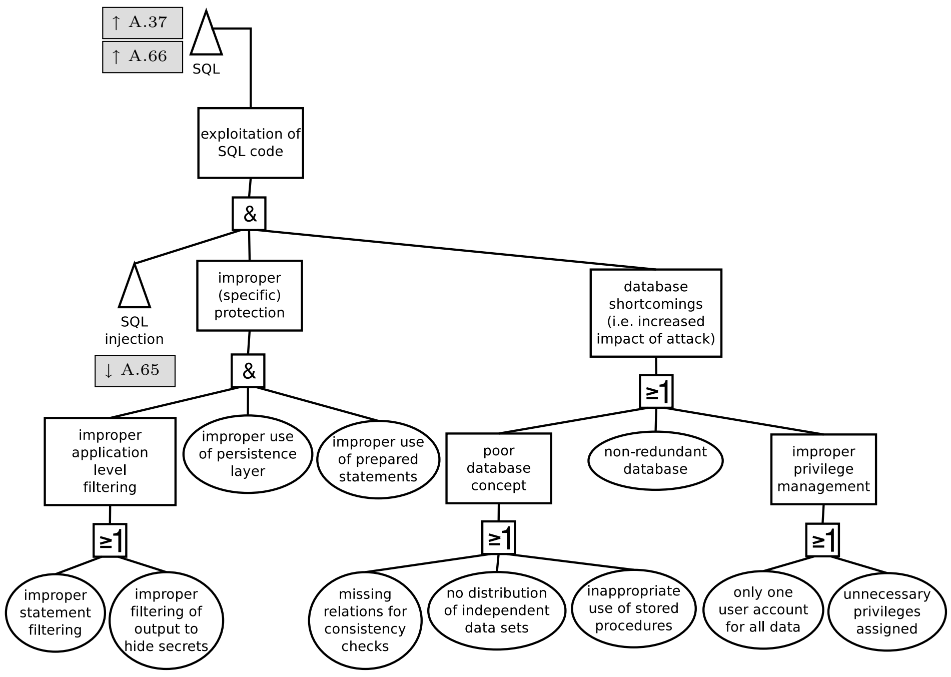

↑ A.37
↑ A.66
SQL
exploitation of SQL code
&
SQL injection
↓ A.65
improper (specific) protection
database shortcomings (i.e. increased impact of attack)
&
≥1
improper application level filtering
improper use of persistence layer
improper use of prepared statements
poor database concept
non-redundant database
improper privilege management
≥1
≥1
≥1
improper statement filtering
improper filtering of output to hide secrets
missing relations for consistency checks
no distribution of independent data sets
inappropriate use of stored procedures
only one user account for all data
unnecessary privileges assigned

Figure A.64: Exploitation of SQL code

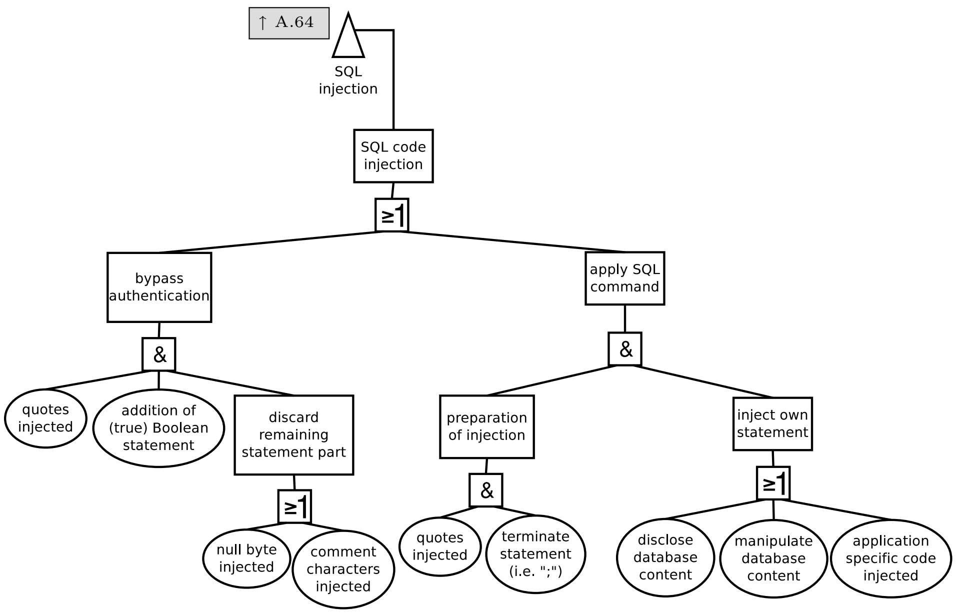

↑ A.64
SQL injection
SQL code injection
≥1
bypass authentication
&
quotes injected
addition of (true) Boolean statement
discard remaining statement part
≥1
null byte injected
comment characters injected
apply SQL command
&
preparation of injection
&
quotes injected
terminate statement (i.e. ";")
inject own statement
≥1
disclose database content
manipulate database content
application specific code injected

Figure A.65: SQL code injection

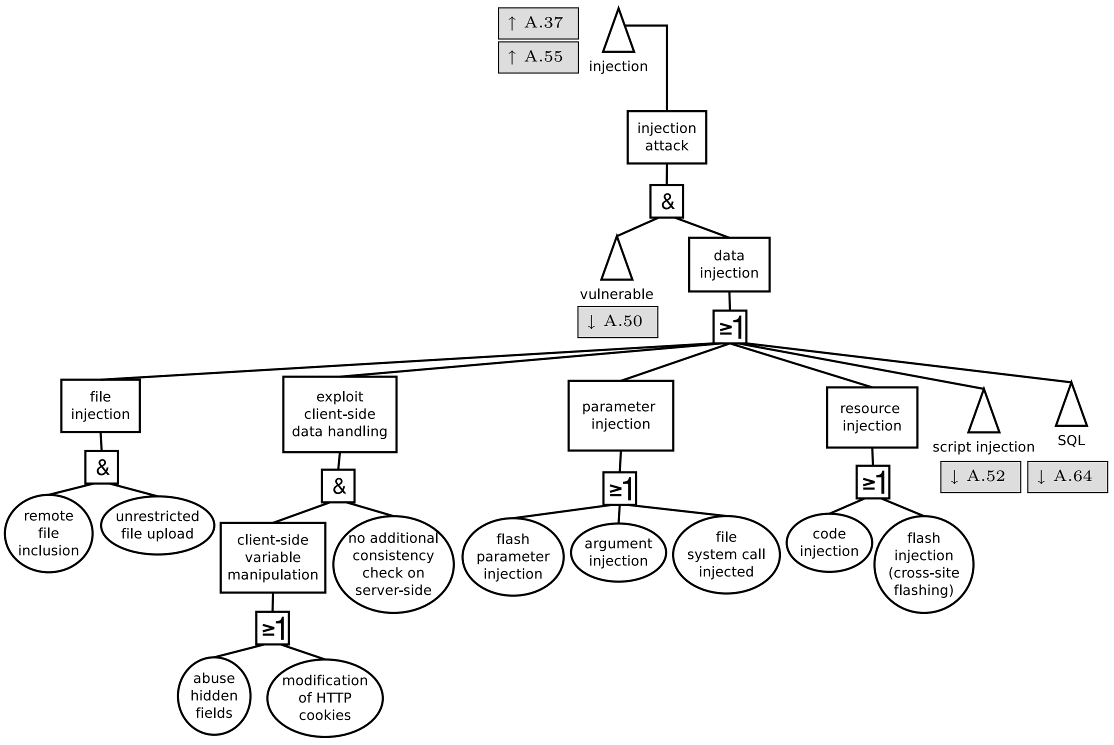


Figure A.66: Injection attack

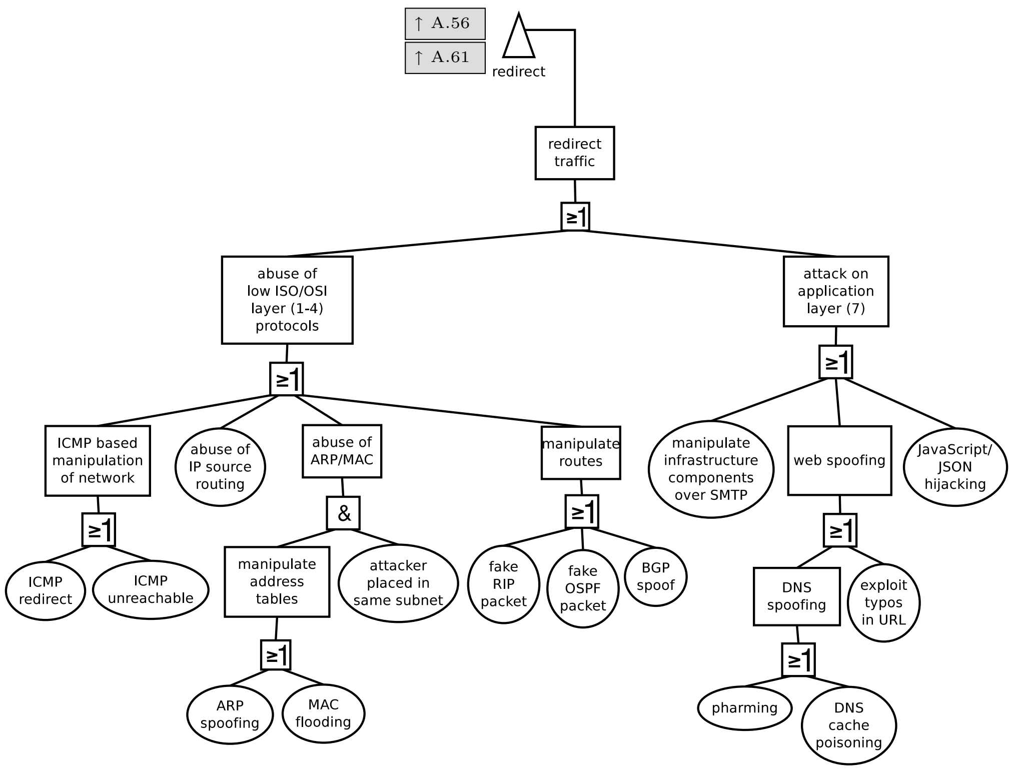

↑ A.56
↑ A.61
redirect
redirect traffic
≥1
abuse of low ISO/OSI layer (1-4) protocols
attack on application layer (7)
≥1
ICMP based manipulation of network
abuse of IP source routing
abuse of ARP/MAC
manipulate routes
≥1
ICMP redirect
ICMP unreachable
&
manipulate address tables
attacker placed in same subnet
≥1
ARP spoofing
MAC flooding
≥1
fake RIP packet
fake OSPF packet
BGP spoof
≥1
manipulate infrastructure components over SMTP
web spoofing
JavaScript/ JSON hijacking
≥1
DNS spoofing
exploit typos in URL
≥1
pharming
DNS cache poisoning

Figure A.67: Redirect traffic