Attack Trees Describing Security in Distributed Internet-Enabled Metrology

Jeanne H. Espedalen

Masteroppgave
Master i informasjonssikkerhet
30 ECTS
Institutt for informatikk og medieteknikk
Høyskolen i Gjøvik, 2007

Institutt for
informatikk og medieteknikk
Høyskolen i Gjøvik
Postboks 191
2802 Gjøvik

Department of Computer Science
and Media Technology
Gjøvik University College
Box 191
N-2802 Gjøvik
Norway

Abstract

The goal of this master thesis was to perform investigation of the attack tree method, and to study some distinctive characteristics of this method. A main objective was to demonstrate how the attack tree method could be applied as a means for analyzing the security level in a specific system.

We implemented the method in a security analysis for a specific system application, iMet, which is a system developed at Justervesenet to facilitate Internet-enabled metrology tasks. Different security issues related to confidentiality, integrity and availability of the system was addressed.

In the analysis we identified critical assets and performed an analysis using the attack tree method, resulting in identification of several vulnerabilities to the system. Following this, possible threats to the system were identified and assessments of risk for the vulnerabilities were performed.

Based on this assessment some possible countermeasures, aimed to mitigate the vulnerabilities, could be recommended.

The attack tree method showed to be suitable for the purpose defined here. The flexibility in this semi-formal method made it possible to perform an analysis with valuable results even for an analysis with limitations in available resources and information.

Sammendrag

Målsetningen med denne oppgaven har vært å studere angrepstre-metodikken, og noen spesielle egenskaper med denne. Et hovedmål har vært å demonstrere hvordan angrepstrær kan brukes for å analysere sikkerhet i et konkret system.

Metoden ble implementert i en sikkerhetsanalyse for et system, iMet, som er utviklet hos Justervesenet for en applikasjon som skal brukes til Internett-støttet kalibrering av instrumenter hos kunder.

I analysen ble forskjellige sikkerhetsrelaterte forhold knyttet til konfidensialitet, integritet og tilgjengelighet undersøkt.

Analysen startet med å bestemme kritiske aktiva og verdier som skal beskyttes i systemet. Angrepstre-analyse for disse gjorde det mulig å identifisere sårbarheter i systemet. Etter så å vurdere mulige trusler for systemet, kunne det estimeres risiko for disse sårbarhetene.

Basert på denne risikoanalysen har det vært mulig å foreslå kontrete mekanismer og strategier for å øke sikkerheten i iMet-systemet.

Angrepstre-metodikken har vist seg å være godt egnet for en analyse som denne. Fleksibiliteten i denne semi-formelle metoden gjorde det mulig å oppnå verdifulle resultater, selv for en analyse med konkrete begrensninger når det gjelder tilgjengelige ressurser og informasjon.

Preface

In this thesis I have had the opportunity to combine my background in metrology and electronics measurements with a study in information security, specifically the study of the attack tree method.

The work has been both interesting and challenging, bringing together experiences and qualifications acquired in studies and professional life.

Not the least challenging has been the combination of studies, work and family life in this period of time. The people in my close surroundings have of course to a great extent been affected by my prioritizing of time, and I will use this opportunity to thank both family and colleagues at Justervesenet for patiently supporting me in this period. Without your encouragement and assistance, this work would not have been possible to finish.

Especially, I will thank my colleague Åsmund Sand for all support and valuable discussions in the matter of understanding the details in the iMet system.

My supervisor, Dr. Stephen D. Wolthusen, who never lost his faith in me, has been my main encourager. His valuable and pointed suggestions and advisory remarks have been essential for my possibility to accomplish what I have done in this work.

Jeanne H. Espedalen, 2007/06/30

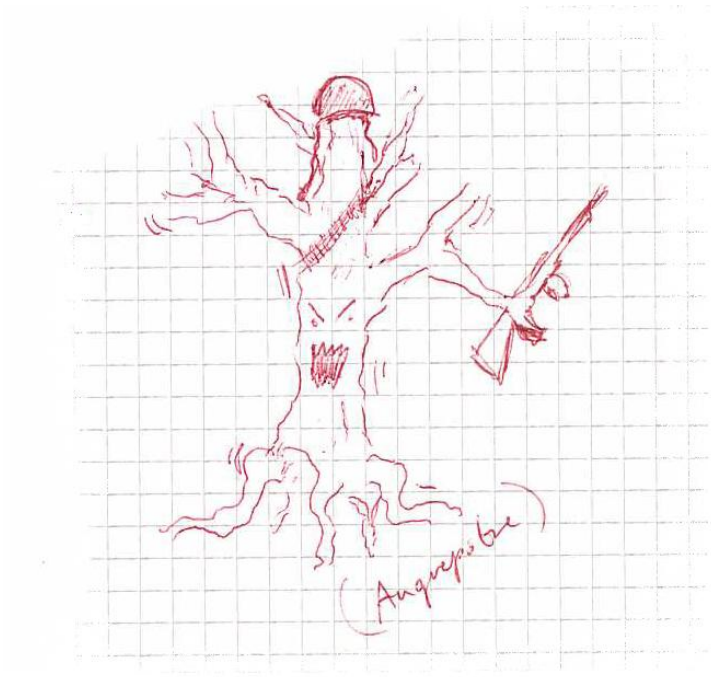

Figure 1: An alternative understanding of an attack tree, by Stian S. Hoem at Justervesenet

Contents

List of Figures

List of Tables

1 Introduction

1.1 Topics Covered by this Thesis - a Reader's Guide

The author of this thesis has background in metrology[1], and has been working in this field, with calibration of electrical measuring instruments, for 17 years. The metrology background has been combined with studies in information security, and in this thesis work it has been a natural choice to combine the interest and experience in those two fields.

Motivation for the thesis work is therefore two-sided. One of the main topics has been the investigation of the attack tree method, and the study of some distinctive characteristics of this method. The attack tree method is based on the description and visualization of possible attacks to a system, expressed in tree structures. An example of an attack tree is shown in Figure 2. We will come back to description of the attack trees, how they are built and how they are used in sections 2.1.3 and 3.1.

A main objective for the thesis work has been to show how the attack tree method could be applied as a means for analyzing the security level in a system, and the evaluation of the usability of the method has been one of the main results of this work.

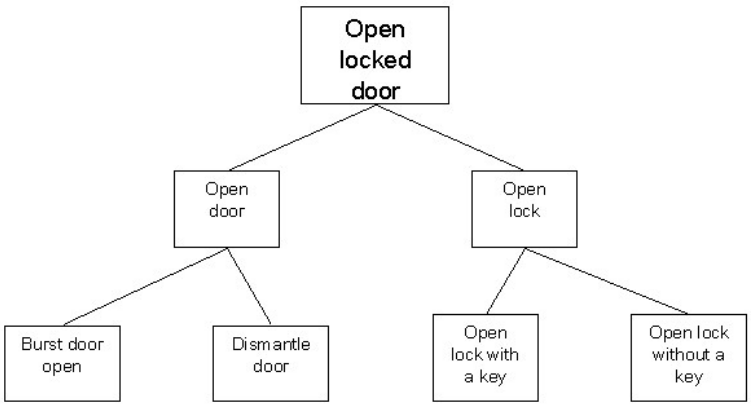

Figure 2: A small example of an attack tree, showing different ways an attacker could enter a locked door.

In section 2.1.1 the attack tree method is presented in relation to risk analysis in a more general context, and in section 2.1.3 a rationale for the selection of this method is given. Section 3.1 presents the attack tree method more thoroughly, and here are also presented other scientific works dealing with attack trees, several aspects with the method, and also some works on related methods and tools.

[1]Metrology, "the science of measurement, embracing both experimental and theoretical determinations at any level of uncertainty in any field of science and technology." as defined by the International Bureau of Weights and Measures, BIPM

The other main objective of this thesis has been the evaluation of the security risks in a distributed Internet-enabled metrology system application. Distributed Internet-enabled measurements, and specifically the development, trends and challenges in the area of distributed Internet-enabled metrology is presented more thoroughly in section 2.2 and 2.3. The specific system to be analyzed, iMet, is introduced in section 2.4, and presented in further detail in chapter 4. Other scientific works dealing with distributed Internet-enabled measurements, and specifically metrology are presented in section 3.2.

Some constraints and restrictions have unavoidably been put on this thesis, and how the work could be presented. The author works at Justervesenet, the organization that owns the system to be investigated, and which is also the organization that will be one of the users of the system in a metrological applications setup. Justervesenet's customers will be the other users, and for both Justervesenet and customers, it would be of vital importance that sensitive information should not be disclosed. Due to this consideration several limitations have been defined. Especially information about technical details concerning network configuration, security mechanisms installed etc., have had to be protected.

The total extent and depth of the analysis has also been restricted by time and resources available for the author. A full depth, detailed analysis of all security aspects of the system, including aspects of related systems that could introduce security risks, would be a quite extensive exercise. Such an analysis would also necessarily have to include investigation in several parts of the organizations involved, and would to a great extent have to involve personnel like system administrators and others with specific competence. The choice has therefore been to focus on security aspects that are considered to be of special interest and importance to the field of distributed Internet-enabled metrology, and to applications in this specific area.

The limitations for the security analysis is where appropriate pointed out together with description of the iMet system presented in chapter 4. Specifically assumptions and limitations that had to be made prior to the analysis is presendted in section 4.6. Here are also some generalizing assumptions presented, taking care of the requirement to protect the confidentiality of system and organization-specific information details.

The security analysis starts out with a list of assets to be protected. These assets are presented in section 4.8. The analysis itself begins with a high-level functional analysis, presented in chapter 5. Based on the criteria of risk level and business criticality, priorities are assigned to the different areas of this high-level trees, and following this, in chapter 6, in-depth analysis of selected areas are presented. Vulnerabilities are identified and presented in chapter 7. The identified vulnerabilities are examined and associated with possible threats to the system. On the basis of these evaluations, recommendation for mitigation strategies are presented in chapter 8.

An overall evaluation of the usability of the method, for an analysis of this system, is laid out in chapter 9, followed by some conclusive statements and suggestions for how this thesis work could be used as a foundation for further studies in chapter 10.

2 Background

2.1 Need for Risk and Threat Analysis

When we enter a car to drive to work, or a plane to go on a holiday trip, we are all aware that there are risks attached to this. The decision to enter a situation that induces us to risk, are based on informal risk and threat analysis. Most of us will for example be hesitant to go on a holiday to countries connected with crime or terror. The 'value' of the trip would be smaller than the probable cost induced. These kinds of risk analysis are based on our own references, experiences and personality. Some of us are more willing to take risks, and the value of the tasks that are evaluated would be quite different for different persons and in different settings. This explains why some of us in fact are willing to take the risk of jumping out from an airplane, with only some fabric and strings packed in a ruck-sack to get us 'safe' down to earth, the 'value' of a life seems to be different, or the risk is determined differently.

For our professional lives, these risk evaluations should be based on more well founded methods. The impacts of bad decisions, attacks leading to loss of property, disclosure of confidential information or loss of reputation could be very unfavourable and even devastating to companies. On the other hand, to be competitive, businesses cannot use unlimited resources to secure systems and processes.

Historically, the challenge to evaluate risks has been based on statistics generated from experienced accidents or 'misfortune'. Unfortunately, risk level evaluation only based on experienced events would not be adequate, particularly when considering risk associated with hostile attacks, either directly targeted at your business or more generally directed at a part of the society, infrastructure etc. The need to understand these threats and how they complicate the picture is fundamental.

Risk and threat analysis is used as a means to establish the level of security in an increasing variety of businesses, and for systems in different disciplines. The need to balance cost with benefit or value when implementing security measures, is one of the drives for the use of such analysis. For information and communication technology systems, as for other disciplines as well, the increasing complexity in applications and systems also means that the users and owners of the systems have to face a more complex picture when it comes to being confident that their systems have a reasonable satisfying security level. The complexity of the situation also makes security analysis a more and more challenging exercise. This is one of the reasons for the development of several methods to support these kinds of analysis, and a growth in the market for supporting tools.

2.1.1 Approaches to Risk Analysis Strategy

The ISO/IEC TR 13335-3:1998 technical report, "Information technology - Guidelines for the management of IT Security" [1], recommends techniques for the management of IT security. It present guidelines for the assessment of security requirements and risks, and advice for the selection of appropriate security safeguards to reach the appropriate security level.

The report presents the different stages in this process and describes different risk

analysis strategies. 4 different approaches for a corporate risk analysis strategy are established:

- Using the **Baseline Approach** implies selecting standard safeguards to all IT systems without doing any differentiation when it comes to the variation in security risk level or criticality. The baseline security level could vary. Selecting a high level baseline would be expensive, and a low level baseline means increased probability for security breach.
- The **Informal Approach** is based on informal and pragmatic risk analysis. The efficiency of this approach would to a great extent be dependent on the knowledge and experience of individuals.
- A **Detailed Risk Analysis** approach implies conducting detailed risk analysis for all IT systems in the organization. This means doing in-depth valuation of all assets, fully assessment of threats to those assets and a following risk assessment including the identification of all necessary safeguards.
- The **Combined Approach** starts with a high level risk analysis for all systems, focusing on the identification of IT systems with high value for the organization and/or exposed to high risk levels. For the identified high value / high risk level systems, a detailed risk analysis is performed. For all other systems a baseline approach is chosen.

For this thesis work, the idea of a combined approach is adopted. The combination of a high level risk analysis for the overall system, and a more thorough analysis for some chosen processes, will make it possible to distinguish between critical and metrological-specific parts of the system for more detailed analysis, and a high-level analysis for the parts that are less critical or less interesting in this context. This approach will possibly also show the flexibility in the attack tree method.

2.1.2 Methods and Tools for Risk Analysis

The different methods and tools that are used to support analysis, varies from more or less systematized routines based on checklists (this is often used in revision of systems), to the employment of more formalized methods to support qualitative and/or quantitative analysis. The usability and efficiency of the different types of methods would vary a lot depending on properties of the systems that are analyzed, and available resources and information. Expertise in using the methods would also be of vital importance.

Methods for analyzing risk or threat could be divided into three main types;

- **Check lists** are often used as part of some kind of routine activity, and could often be compared to revision of systems or businesses. The quality of the analysis depends to a great extent on how the lists are adapted to the system or the items that are analyzed, and how the results are processed further. For complex systems with interdependencies between different modules and functions it would be very difficult to effectively use checklist-based methods.
- **Qualitative analysis** are often based on the filling-out of questionnaires or forms. The analysis is based on experts and users describing and identifying assets to protect. The possibility of unwanted incidents are in some way quantified or given some kind of probability or classification. Consequences of unwanted incidents could also be quan-

tified, (as loss of money, human life, reputation etc.). This quantification would be based on subjective evaluation and for these kinds of analysis, no objective measurements. Results from the analysis are often presented in risk matrices, and choices and prioritizing of countermeasures would be based on how the acceptable risk levels are introduced into these matrices and how the overall picture is interpreted. The concept of qualitative analysis is also the basis for the attack tree modeling method and other similar concepts as for example fault tree modeling, described in [2]and [3], and flaw hypothesis analysis, introduced in [4].
A security risk analysis using a qualitative analysis will be a fundamentally creative process. The quality of the results of such an analysis would have to be based to a great extent on skills and experience of the person(s) performing the analysis. But the usability of the method chosen, for the specific system being analyzed, would also be vital.

- **Quantitative analysis** would in some way have to be founded on formalized methods, and the methods would include some kind of objective measurements using collected data. The results of the analysis depend to a great extent on experienced data and the completeness and quality of this data. This would normally indicate that an analysis based on quantitative method requires quite extensive resources. Quantitative analysis could, when appropriate, be used as a part of a qualitative analysis, to support the more overall qualitative evaluation of a risk level.

In Table 1, a comparison of some of the characteristic of these methods is made.

For information and communication technology systems, several formalized methods and tools have been introduced and used for some years. ASIS NORWAY, committee for risk analysis, presented in 2003 a report, [5], discussing some of the most used methods and tools for risk analysis in Norwegian businesses. Some of the methods and tools are listed as examples in Table 1.

2.1.3 Why Attack Trees?

The attack tree method approach, which is discussed further in chapter 3.1, is to define and analyze the different attack pathways, for e.g. computer systems and networks, in a structured way. This is done by modelling trees, defining the goal of the attacks as roots and refining the model by describing possible attack paths through child nodes. Leafs of the trees represent attacks that no longer can be refined. This basic attack tree mechanism proposed by Schneier [6], could also be extended using conjunctive (AND) or disjunctive (OR) refinement at the child nodes. The nodes could also be associated with attributes describing cost, effort, motivation or other necessary resources essential to an attack. These would be limiting factors for attackers.

One of the obvious advantages with the attack tree approach would be that once you have a picture of the possible attacks, the leaf nodes could be extended with attributes e.g. cost levels or other capability or behavioural indicators. This picture shows possible vulnerabilities in the system and with attributes, possible threats, resulting in a visualized representation of risk. Following this it would be quite straightforward to extract the most effective mitigation strategies - to insert protective measures to cut off the attack that implies lowest cost for the attacker (- or the lowest skills, highest motivation etc).

The attack tree method could be used in all phases of the development of a defined system. It will also be possible to perform the analysis for a system in specifically de-

Characteristics	**Check lists**	**Qualitative**	**Quantitative**
Working method, information gathering	Filling out forms	Questionnaires Information gathering Brainstorming	Objective measurements Sets of experienced data Statistics
Analysis	Compliance/ noncompliance	Evaluation Prioritizing	Calculations
Examples of methods or tools - standards	ISAP - ISO/IEC 17799 - ISO/IEC 27001	SARA - ISF Standard of Good Practice NSO - NS 5814	KvantRisk - NS 5814 for threath identification
Requirements of personnel involved	Both qualified/ nonqualified	Both qualified/ nonqualified	Qualified
Presentation of results	Compliance/ noncompliance with security profile or minimum requirements	Risk matrices	Probabilities Money Percentages Cost/benefit
Problems	Are the real threats to a system assimilated ‘behind’ the check points Prioritizing	Subjective evaluation	Validation of data sets Complex calculation

Table 1: Characteristics of different Risk Analysis Methods

fined contexts and environments, as long as the outline and interfaces are defined. The structure of an attack tree is expressed in a node hierarchy. This makes it feasible to split an expressed abstract attack on an attack goal into several concrete sub-attacks, which are aimed at sub-goals. This decomposition perspective together with the possibility to define selected areas for analysis, makes it possible to design and analyze attack trees at different levels and for selected areas of interest.

This possibility to decompose attacks, and describe and analyze these on different levels of detail and abstraction, would be used in our work. The idea is first to do a broad analysis of 'main' attack goals and possible attacks on these. Then we will choose specific areas of interest and dig into these in more detail. This would be done in an iterative fashion, getting us in steps closer to a complete picture of possible attacks in the selected areas.

In this way one could combine an overall picture of possible attacks and treats to the system, with a more in-depth investigation of chosen areas of interest. These areas could be picked based on special interest or due to expectations of vulnerability.

A somewhat similar analyze method, flaw hypothesis testing, is also based on propagation of errors due to vulnerabilities, with the aim to find possible ways to attack a system. The method was first introduced by Linde in [4], and the idea is to hypothesize possible flaws, and then test whether these hypotheses are true. The method is also called penetration testing when applied to security testing, and is used in several tools specially built for network security testing, e.g. SAINT, [7].

To effectively do flaw hypothesis testing, or penetration testing, the quality of data available (collected from network traffic, vulnerability databases) will be of vital importance. The formalistic basis for the method implies that a test run has to be extensive and take into account every flaw, node etc. Thus not leaving any possibility for improvement of efficiency by taking 'shortcuts' based on more creative decisions. A problem, for a system owner using these tools, would be that the presentation of the results, principally TRUE and FALSE for tested flaws, gives no indication of values of the flaws giving TRUE result or any kind of prioritizing of which one to act upon.

The mechanism of attack trees could be defined as semi-formal, as the systematic approach is complemented by the creative nature of modelling possible attacks. Associating attributes, and giving those attributes values, could be done using both quantitative and formal methods or by evaluation and using more creative measures. One of the goals of this thesis would be to explore how this flexibility could be employed and taken advantage of in a described scenario.

2.2 Distributed Internet-enabled Measurements

Today, the Internet has become a much used communication medium for performing or assisting measurements[1] made in locations geographically parted from where the operator or controlling system of these measurements is. Applications and systems are used in different technology areas, for both commercial and non-commercial use. Examples are telemedicine, education, traffic surveillance, air pollution surveillance, distributed laboratory systems, etc.

The degree of remote operation in these systems varies a lot, from simply monitoring

[1]Measurement, is defined in International Vocabulary of basic and General Terms in Metrology , [8], as "set of operations having the object of determining a value of a quantity"

signals or systems using sensors or web-cameras (e.g. surveillance), to guiding operators on site (e.g. education) or having full control of measurement set-ups and handling measurement data from a remote location.

2.3 Distributed Internet-enabled Metrology

In metrology, traceable[2] calibration[3] of an instrument is traditionally carried out by sending the unit under test (UUT) to a reference laboratory. Here, the UUT is calibrated against a standard, which again is traceable to a national standard. The calibration is performed in a controlled environment, often with an automated measurement procedure and skilled personnel performing the measurement set-up.

The instruments that have to be calibrated are often expensive and much used, and many businesses have problems with long down-times when expensive equipment is to be calibrated. Transportation of the instruments also adds uncertainty to the calibration results, and there will always be a risk that instruments could be broken or have reduced quality due to improper handling during transport.

The spread of the Internet as a communication medium and the availability of high precision travelling measurement systems, open up an opportunity to disseminate calibration values from national standards laboratories such as Justervesenet [9] in a different way.

In the international metrology society there has been a trend the latest years toward providing remote operation over the Internet to more and more calibration and measurement services. Several coordination projects have been and are addressing this in different ways.

The European thematic network "SofTools - MetroNet" [10] resulted in several conferences and meetings where topics regarding distributed Internet-enabled metrology were discussed and several works were presented.

Several ongoing activities internationally are addressing Internet-enabled metrology, and this is further addressed in chapter 3.2. The main idea in most of these works, as it is for the iMet-system described more thoroughly in chapter 4, is that the unit under test (UUT) is not moved from the owner's laboratory. Instead a calibrated standard, e.g. an electrical multimeter, is transferred to the laboratory from a reference laboratory (e.g. an national measurement institute, NMI), and the calibration process is then controlled directly by the reference laboratory.

There are many advantages to this approach. Since the UUT is never moved, the uncertainty associated with the transport of the UUT is eliminated. The reference laboratory often has much better understanding of the transport uncertainty of the calibrated standard, so the uncertainties due to transportation will presumably be smaller than for the traditional system. The calibrations are performed in the owner's environment ensuring that measurement results accurately reflect the conditions relevant to the owner's situa-

[2]Traceable (calibration), Traceability means that the result of a measurement, no matter where it is made, can be related to a national or international measurement standard, and that this relationship is documented. In addition, the measuring instrument must be calibrated by a measurement standard that is itself traceable. Traceability is thus defined as the property of the result of a measurement or the value of a standard whereby it can be related to stated references, usually national or international, through an unbroken chain of comparisons all having stated uncertainties, International Bureau of Weights and Measures (BIPM)

[3]Calibration, is defined in International Vocabulary of basic and General Terms in Metrology, [8], as "set of operations that establish, under specified conditions, the relationship between values of quantities indicated by a measuring instrument or measuring system.."

tion. Transportation cost could also be reduced, especially when using a scheme where the standard is transported to several customers successively.

The need for reliable and secure applications and solutions has been expressed in different contexts. The EUROMET[4] work group "Math and Software" expresses this in a Roadmap: "Towards an IT based Metrology", shown in Figure 3. The Road Map was developed and harmonised in the EUROMET focus group "Math and IT" for future activities, at a meeting in Berlin 16. og 17. februar 2006. In this roadmap one of the main targets was expressed as "Reliable and trustworthy IT solutions for metrology". And the metrological applications necessary to reach this are among others "“Remote calibration and testing"' and "“Integration of security into metrological systems"'.

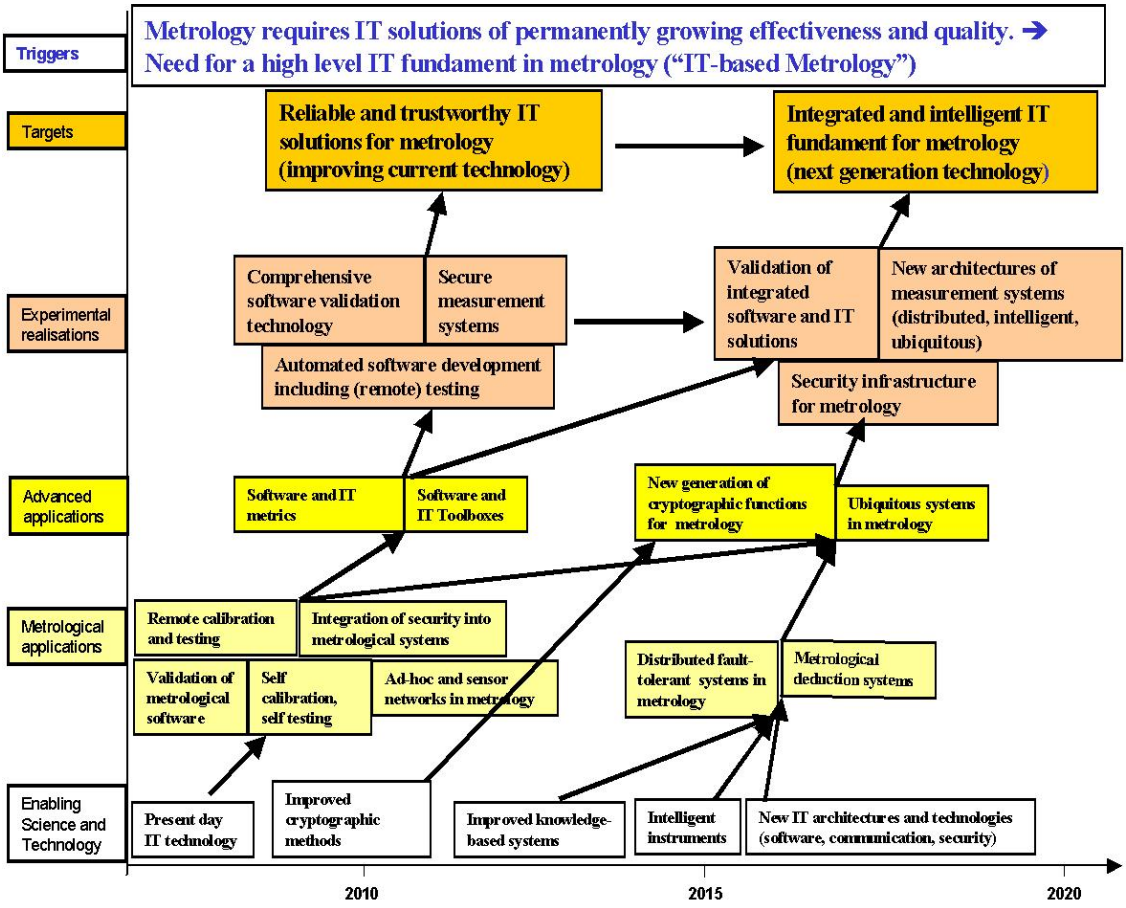

Figure 3: The EUROMET roadmap: "Towards an IT based Metrology", developed and harmonized in the EUROMET focus group "Math and IT" in Berlin, February 2006

In the ongoing EU funded project "iMERA" [11], which started in April 2005 and will last for 3 years, one of the tasks deals with "Understanding information and communication technology tools", and here "data confidence and security" is one of the main issues.

Calibration and test laboratories, offering calibration services, are accredited by the NS-EN ISO/IEC 17025 standard, [12]. This standard states requirements regarding the

[4]EUROMET is a cooperative voluntary organisation between national metrology institutes (NMIs) in the EU and EFTA, including the European Commission. See http://www.euromet.org/

calibration procedure, documentation and competence for parties involved in the calibration process. An accreditation of a calibration service will require documentation of adequate competence and quality control of the calibration process and results. For a service based on Interned-enabled calibration to be accredited, it will be expected that much focus will be on how the process is controlled and documented, and also how the integrity and confidentiality of the calibration results are taken care of. These issues are very much related to and depending on information security issues for the system used in the calibration process. Therefore, a security assessment for this system will be of great importance for an accreditation of a calibration service based on the system.

2.4 Distributed Internet-enabled Metrology at Justervesenet, iMet

At Justervesenet, Åsmund Sand has developed a system, iMet [13], which enables a person at one local area network (LAN) to control and monitor the instruments located at another LAN, as if these were connected locally at the former. The system is offering a bi-directional communication link between two PCs, located at different LANs separated by the Internet. A viable application for this system is Internet-enabled traceable calibration, where a measurement standard is sent to the customer's laboratory, and the calibration authority's laboratory (Justervesenet) controls and monitors the calibration process directly at a customer's laboratory.

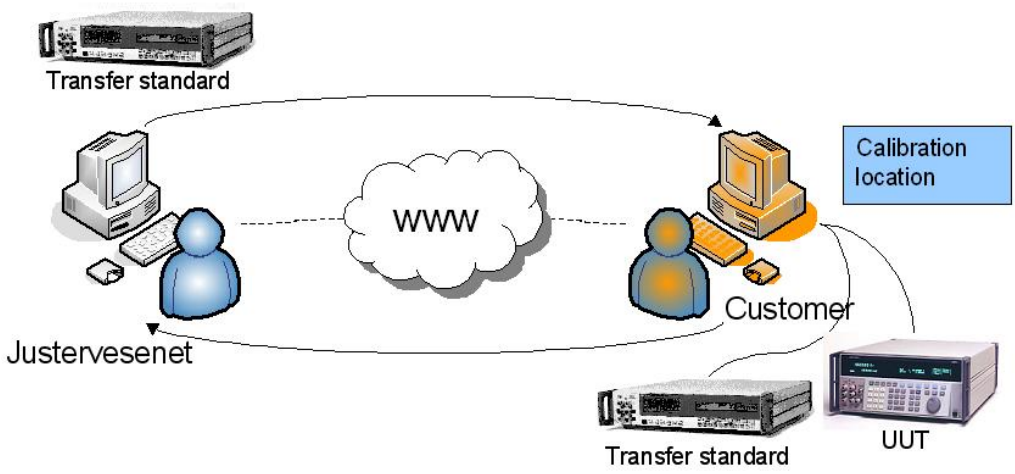

Figure 4: Calibration of customer's instruments using the Internet to control and monitor the process directly at the customers laboratories

The iMet system provides a flexible and general solution for an Internet-enabled application, and has been designed to support current and future requirements for flexibility in network architecture. In contrast with most of the ongoing activities in other countries, the iMet system is designed to support different metrological activities and not one specific service. Several security concerns have been addressed in the design and implementation of the system, but for customers and accreditation authorities, an assessment of the security level would be of vital importance.

2.5 Motivation and Benefits

Up till now, the main focus in most of the ongoing activities in the area of distributed Internet-enabled metrology has been on functionality of the systems. Security issues are addressed in some projects (see chapter 3.2), but there is still a lack of understanding of several of the security challenges that should be addressed.

The demonstration of an assessment process of a system designed for distributed Internet-enabled metrology, should give valuable information on how such a system should be securely implemented. This assessment should comprise an overall evaluation of different security concerns generally applicable for distributed Internet-enabled metrology systems, and the results of the process would hopefully give a better understanding of threats and vulnerabilities in such systems in general.

The goal of this master thesis will be to point out security related challenges, both general and more system-specific. But it is also expected that the results should contribute to the foundation of best-practice guidelines and general advice on how to implement secure systems. In this way the results could contribute to the general acceptance of Internet-enabled metrology as a means of dissemination of calibration values.

For the iMet system, the assessment would lead to a better understanding of security concerns related to this specific system, and suggest feasible mitigation strategies for the system.

By using the attack trees method to describe and evaluate security issues in this system, we are also hoping to contribute to the understanding of suitability of this method in security assessment. In particular we hope to show how this method could provide an understanding based on cost-benefit evaluation of where to put the effort to enhance security in the system.

3 Related Work

We have studied several other scientific works related to both the attack trees methodology and distributed Internet-enabled measurements. In this chapter we present summaries of these works and we also draw some lines to the work in our thesis.

3.1 Attack Trees

In this section we first present the basic concepts of the attack tree methodology, then discuss some works where the concept is taken a bit further into a wider abstraction, attack graphs. We also present some works where the methodology has been applied in security analysis.

3.1.1 The Concept of Attack Trees

The concept of attack trees is derived from fault trees in software safety. Fault trees are used to describe how errors propagate in software systems, and analysis of this could be used to test software, as described by Leveson in [2] and Viega and McGraw in [3]. Whereas fault trees are most commonly used to model how problems occur in critical systems, given the built in focus on error propagation, attack trees have a slightly different viewpoint. The introduction of an attacker, or group of attackers, makes it possible to model threats to a system including the aspect of targeted attacks. Considerations of probabilities based on available money, tools or motivation for the attacker, gives a more thoroughly grounded picture of the risk level.

Bruce Schneier presented the concept of attack trees as a way to model threats against computer systems, in an article in the Dr. Dobb's Journal in 1999, [6]. The idea of the concept is to understand the threats to a system by describing all the different ways in which a system can be attacked and compromised, in a tree structure. The root node in the tree representing the goal for an attacker, or the value or asset to be protected, and the leaf nodes representing the different ways of achieving that goal. Based on the tree structures and information of who the attackers are and which capabilities they possess, we will then be able to more effectively install the proper countermeasures and to deal with the real threats.

The basic tree model presented in Schneier's article describes the Attack Tree with two different types of nodes, AND-nodes and OR-nodes. At OR-nodes a minimum of one sub-goal should be satisfied to achieve the goal of the node. At AND-nodes every sub-goal should be satisfied to achieve the goal of the node. Values can be assigned to leaf nodes, like boolean statements as possible/impossible or special equipment required/special equipment not required. Boolean calculation could be performed on the values of sub-goal nodes based on the boolean expression (AND-/OR) in the node, giving a resulting statement in the node. An example tree showing the basic principles is shown in Figure 5, the example shows an attack tree for a physical attack against a locked door. OR expressions are implemented in the tree by linking the sub-goals together to the resulting node. AND expressions are implemented by and arc connection together with an "and" between the links from the sub-goals.

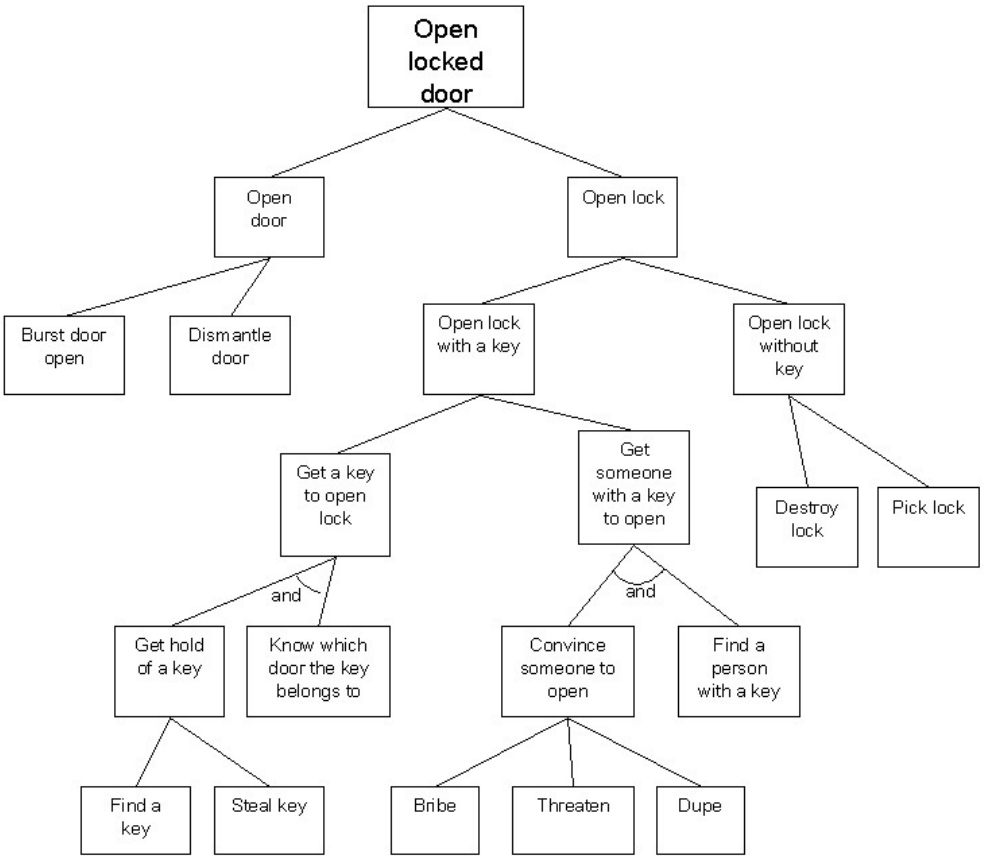

Figure 5: An example of an attack tree, showing different ways an attacker could enter a locked door. OR expressions are implemented by linking the sub-goals together to the resulting node. AND expressions are implemented by and arc connection, together with an "and", between the links from the sub-goals.

Non-boolean values indicating the effort of an attack (cost or time) could also be assigned to a node in Schneier's model, but calculations on these values are limited to summation of sub-node values for AND-nodes and selecting the cheapest sub-node value for OR-nodes.

In [14], Moore et al. describes a means for documenting information-security attacks in a structured and reusable form, using attack trees, and exemplifying this with a fictitious company and with known attacks. The focus is on documenting and organizing generic patters of attacks to be able to reuse these in the attack tree construction, and thus refining the attack tree model with the reuse of generic attack patterns. The report points out several questions about the use of attack trees for documentation of vulnerabilities.

Mauw and Oostdijk presents in [15] a suggestion to formalize the attack tree concepts. The concept of attack suites as a level of abstraction of an attack tree is introduced. Semantics is suggested to make it possible to perform a formal interpretation to understand how attack trees can be manipulated during construction and analysis. The authors argue in the work that implementation of semantics and formalization of the method is necessary to make it possible to effectively implement the method in automated tools.

3.1.2 Attack Graphs

In several works, like [16], [17], [18] and [19], the attack tree concept is taken a step further, and the term attack graphs is introduced (although this is not explicitly defined in any of these works). Amman et al. describes in [20] the two terms like this:

> "..it is useful to think of an attack tree as a structure in which each possible exploit chain ends in a leaf state that satisfies the attackers goal, and an attack graph as a consolidation of the attack tree in which some or all common states are merged"

In [16], Philips et al. presents a graph-based approach to network vulnerability analysis. A model for a tool and a method for analysis of possible attacks are the central elements in their paper. The model is presented theoretically, and describes a scenario that depends on three elements:

- Available input information of *common attacks* systematized in a database,
- Information of network configuration and topology in a *configuration file*
- *Profiles* of possible attackers

The analysis takes a systematic approach and is based on automatic generation of an attack graph. The attack graph is the customization of generic attack templates to the attacker profile and network specified in the configuration file. In the resulting graph, any path is then representing an attack. Each edge has a weight, which could represent the success probability, average time to success, cost/effort level to attacker etc. This weight would be a function of both the system configuration and the attacker profile. The result from analysis of a system could then result in a set of attack paths, with the possibility to focus on low-cost attacks. This could then be used to determine the most cost-effective defences for the system. The writers also argues that the model makes it possible to simulate dynamic attacks, based on information of changes in topology, different levels of attacker abilities etc. In [21] the tool is presented, and implementation issues are discussed. The attack graph is constructed by using an algorithm that is exploring the system from an initial state.

Cohen et al. presents in [22] a classification scheme for information system threats, attacks and defenses as a cause and effect model. Here are also presented some analysis based on that model.

The idea of automated generation of attack graphs is explored in several other works. The concept of privilege graphs is presented by Cacier in [18], and a requires/providers model for attacks is presented by Templeton and Levitt in [17].

Ritchey and Amman, [23], introduces the use of a model checker for vulnerability analysis of networks. Model checking is a technique used to check whether a formal model of a defined system satisfies a given property. When using this to check for security properties in a system, the result could be used to single out probable attacks or attack chains that could lead to violation of security properties. The attack graph shows how an attacker can force the system into an unsafe state. Ritchey and Amman encode the vulnerabilities in a state machine description suitable for a model checker, and then use the model checker to assert that an attacker cannot acquire a given privilege on a given host. The model checker either offers assurance that the assertion is true on the actual network or provides a counterexample detailing each step of a successful attack. One drawback of their method is that one only can obtain one counter-example, so only one attack giving an unsafe state is found in each 'run'.

Sheyner et al. presents in [19] a model for automatic generation of attack graphs including an algorithm and a tool to implement the algorithm. The tool is based on a modified version of a model checker tool. When using this to check for security properties in a system, the result could be used to single out probable attacks that could lead to violation of security properties. The attack graph shows how an attacker can force the system into an unsafe state. A model checker works backwards from the goal state, this gives savings in space because vulnerabilities that are irrelevant to the goal are not investigated (as opposed to the forward method in [16]). Sheyner also argues that using a general modeling language gives the advantage of including any erroneous behavior of the system into the model, not limited to what could be expected as a result of attacks. In [24], Sheyner and Wing presents an attack graph toolkit and examples of the use of the earlier presented algorithms.

In [25], Jha et al presents a formal analysis of Sheyner's model. This work formally proves that the model is exhaustive (includes all attacks) and succinct (address only relevant states).

Amman et al. also uses model checking for vulnerability analysis in [20]. They use the assumption of monotonicity to reduce scalability problems related to attack graphs. This assumption of monotonicity implies that when the preconditions of an exploit first becomes satisfied, then it never becomes unsatisfied (in a stable model of a system).

In another work [26], Amman et al. deals with the scalability problem in a somewhat different way, using the penetration tester perspective and introducing a suboptimal solution. This work uses a host-centric approach, and shows that the task could me made computationally feasible, but the approach means that testing would not explicitly identifying every possible attack.

Mehta et al. attempts in [27] to reduce the scalability problem of attack graphs by identifying relevant portions of an attack graph and proposing a ranking scheme for the states of an attack graph. The ranking of a state is based on factors like the probability of an intruder reaching that state, and the rank shows the importance of the state. With

a ranked attack graph, it would be possible to concentrate on relevant subgraphs to select suitable countermeasures. The authors presents two algorithms to rank states of an attack graph including examples.

The need to effectively represent sophisticated attacks is addressed by Daley et al. in [28]. Here, a methodology for capturing the structure of various network vulnerabilities and multi-stage attacks is suggested. The attack tree paradigm is here extended and used to describe a context sensitive attack modeling framework that supports incident correlation, analysis, and prediction. The framework is described through abstraction, the attack tree structure is enhanced and expressed in a formal manner in order to represent the complexity of sophisticated attacks.

The generation of attack graphs is addressed by Zhang et al. in [29]. They here present a method to generate attack graphs based on analysis of the host computer, link relation to devices and the characteristics of attacks. Thereby a model of network security status is built, using a forward-search, breadth-first and depth-limited algorithm to produce attack route. Some experiments are performed to validate the prototype of the network attack graph generating tools presented.

Noel and Jajodia has in [30], yet another approach for managing attack graph complexity. This work addresses though not the generation of attack graphs, but focuses on interactive visualization for the management of attack graph complexity. By including hierarchical aggregation of graph elements and managing network attack graph complexity through interactive visualization, the framework presented could provide compression of attack graph complexity.

In [31], Wing presents the concept of scenario graphs, representing sets of counterexamples. A counterexample is produced by traditional model checking to illustrate a violation of a property by a model of the system. In the paper it is explained how scenario graphs are a natural representation for attack graphs.

Dantu et al. in [32] addresses risk management related to attacker behavior by the introduction of behavior based attack graphs. A method to estimate the risk level of critical resources being compromised is formulated based on attackers behavior and the representation of possible attack paths in the analyzed system. Based on the graphs it is demonstrated how calculation of the risk level of a critical resource is possible by using Bayesian methodology.

3.1.3 Application of the Attack Tree Method

The work by Fung et al. in [33], has a somewhat similar approach to the one we have in the work presented in this thesis. This paper presents an initial case study of how the attack tree analysis methodology could be adopted for a survivability study. The process described starts out with the identification of the basic service components in the system and the underlying mobile ad-hoc network, next a defined mission objective was defined and attack tree analysis on the model system was conducted. From the attack tree analysis, identified intrusion scenarios were presented and a quantitative measure for system survivability was suggested..

In [34], Gegick and Williams focus on matching attack patterns to security vulnerabilities in software-intensive system designs. By using an abstract representations of attacks such implemented as attack trees and attack nets it is shown how one can identify potential threats before a system is released. Attack patterns visualizing security vulnerabilities

in a software-intensive system design was constructed, and a case study of the approach was presented indicating that attack patterns can provide general descriptions of vulnerabilities.

Another example where attack trees have been applied in security analysis is presented in a work by Higuero et al. [35] The attack tree methodology is used in a system for securing digital contents in e-commerce protocols with copyright protection. The authors stated here validity for the attack tree methodology for studying and analyzing the security of the distributed systems presented, and as a mean to compare the security level of similar schemes.

3.2 Distributed Internet-enabled Metrology

In recent years, quite a few works on Internet-enabled calibration and distributed measurement architectures have been published. Some of these are described by Rayner in [36]. Most of the systems described have a client-server approach where the client connects to a server, which offers access to measurement set-ups. Examples of measurement applications having this approach are presented in [37], [38], [39], [40], [41], [42]and [43].

In some cases the client is offered downloading of procedures to be run locally, this approach is used by NPL [1] in [44] and [45].

Some systems also use protocols for bi-directional communication between the server and client as IEN in [46], but in this work no special effort is made to ensure that this communication will work through all firewalls and proxy servers.

Security issues are addressed in these works to various extents. NPL has presented a report, [47], with guidance for protecting the data that is transmitted over the Internet and stored in calibration databases. This report also includes an example of a security audit for one of NPL's systems [44].

The iMet system, [13], provides a general solution, which is not dependent on the types of network connection involved. The design of this system is aimed to support current and future requirements for flexibility in network architecture. This is the system analyzed in this thesis work, and a thorough description of the system is presented in chapter 4.

[1] NPL - National Physics Laboratory, The UK's National Measurement Laboratory, http://www.npl.co.uk/

4 The iMet System

This chapter includes a detailed description of the iMet system. This description will later be the basis of the security analysis. The following description starts with a thorough functional description which to great extent follows the development of the iMet system explained by Sand in [13]. Further, the different hardware and software components are identified and described. To do this certain assumptions have been made, as the actual system only has been operated for test purposes until now. In the last sections of the chapter, security mechanisms built into the system and different assumptions for an analysis are discussed. We finish the system description with the identification of different assets in the system that should be the objectives for protection in the following analysis.

4.1 Goals and Scope for the iMet System

The iMet system presented in [48] and [13], was designed at Justervesenet by Åsmund Sand. A system application was developed to operate and monitor instruments at remote locations, the application includes functionality to support Internet-enabled calibration. The aim of the project was to implement a system that was both firewall-friendly and secure. As discussed earlier, there exists several works dealing with remote measurement and calibration of instruments using the Internet as a communication channel.

The innovative property of the iMet system, introduced in the first version, is that it lets an operator remotely control an instrument located behind firewalls or proxy servers. This is achieved by setting up a bidirectional communication channel between two hosts separated by the Internet, using specialized middleware software. The details of this channel will bu further discussed in section 4.2.1

Later versions of the iMet system focused on scalability issues. The second version introduces the possibility to download and run measurement procedures when needed. New procedures may be added to the system without system recompilation or restart. The scalability issue is taken even further in the third version, also called the 'Generic Instrumentation System'. This version also supports scalability in terms of hardware components i. e., the possibility to add new instruments to the system at runtime. The aim of this version is to include the possibility to dynamically add support for potentially any new hardware at runtime.

In the following, detailed descriptions of the three versions of the system are shown subsequently. As the system has been elaborated, most of the properties and characteristics of the earlier versions are also included in the later versions. The final system is thus comprised of the different elaborative steps, and the system described in section 4.4 thus includes most of the properties of the earlier versions.

As will be seen in the description of this version though, it is not a fully tested and apparently not a finalized version to all details. The description of the system is more of a description of a somewhat generic framework for Internet-enabled calibration, and the implementation of this system could be done in several different environments and with different settings and components included. For the purpose of this thesis and the security analysis, some assumptions regarding the chosen technology and architecture

for the system will therefore have to be made. This will be described more closely in section 4.6.

4.2 iMet v1

An overview of the first version of the system is shown in Figure 6. The initial idea of a firewall-friendly communication channel was implemented by designing a two-way HTTPS-channel. The two-way channel makes it possible for the NMI and Customer computers to communicate at what appears to be a direct connection between the two. The measurement procedures and data used in the calibration process are stored at the IIS WEB-server, and both the customer computer and the NMI computer might access methods or data at the server.

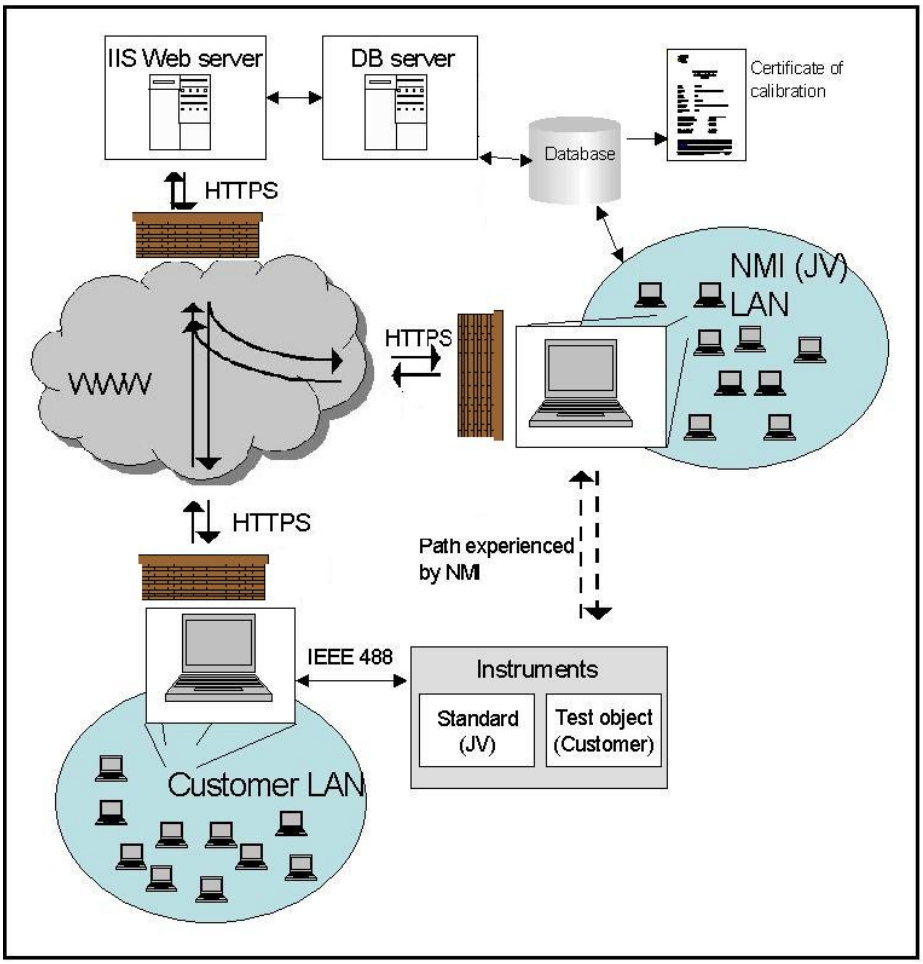

Figure 6: Overview of the iMet system version1

4.2.1 Middleware and full-duplex HTTP(S)-channel

To enable data objects to communicate across the Internet, Microsoft's .NET Remoting technology [49] is used. This is an Object-Oriented-Middleware (OOM) technology, which uses interface-based method invocation over TCP/IP networks. The middleware handles all network related issues in a method invocation process. After a connection to the remote server object is established, the client has access to a local proxy representation of the remote server object. When calling a method on the object proxy, the method call is automatically transferred by the underlying middleware-layer to the real server object. In this way the client is apparently using the remote server object as if it was local. (And when two clients are connected to the same remote server object, communication between them is possible) The communication process transparency is illustrated in Figure 7.

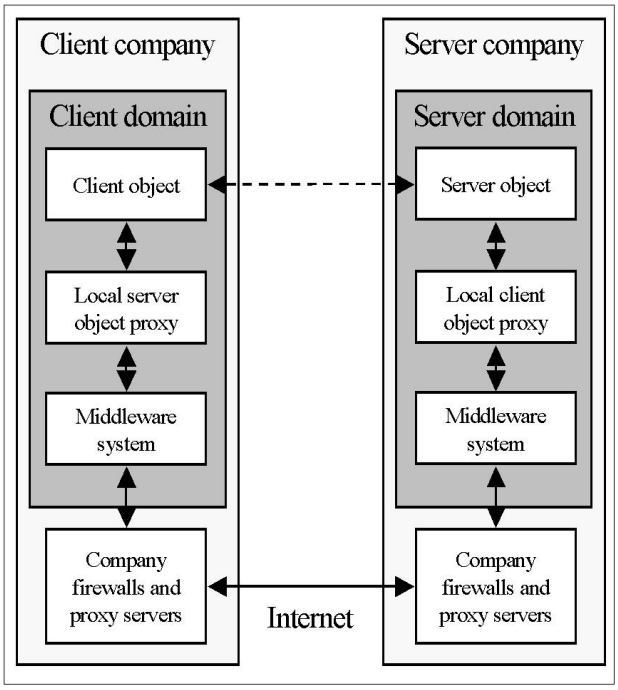

Figure 7: Visualization of the interaction transparency between objects on computers on different LAN's, separated by the Internet. The transparency is achieved using Object-Oriented-Middleware (OOM) technology. The client object and client object proxy implements the same software interface, as for the server object and server object proxy.

The .NET Remoting technology was chosen as the preferred OOM-technology, due to it's flexibility and the possibility it provides to effectively implement a full-duplex HTTPS channel between client and server. The only firewall-configuration required is that port 443(HTTPS) has to be open in the server firewall, which is quite common in a normal business network setup.

The full-duplex HTTPS-channel used in the iMet system is based on a design from GenuineChannels, [50]. The channel design from GenuineChannels is basically a full-duplex HTTP-channel, and the following description of the functionality will refer to HTTP-connections. In section 4.5 we will describe how the channel is implemented as a

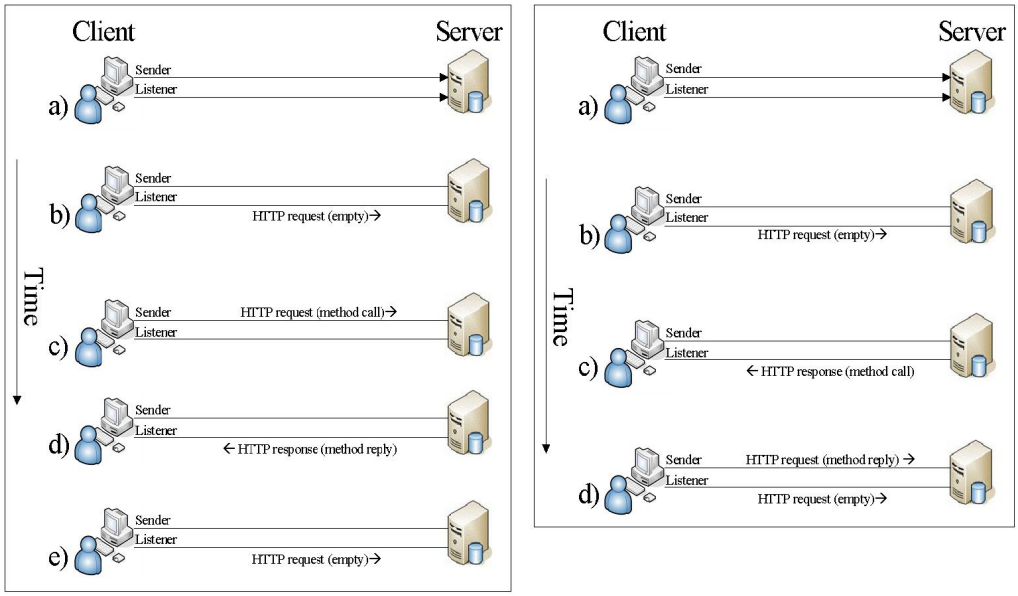

Figure 8: Set up of a two-way HTTP(S)-channel and method invocation. On the left side is illustrated how the a client to server method invocation is performed and at the right is server to client method invocation illustrated

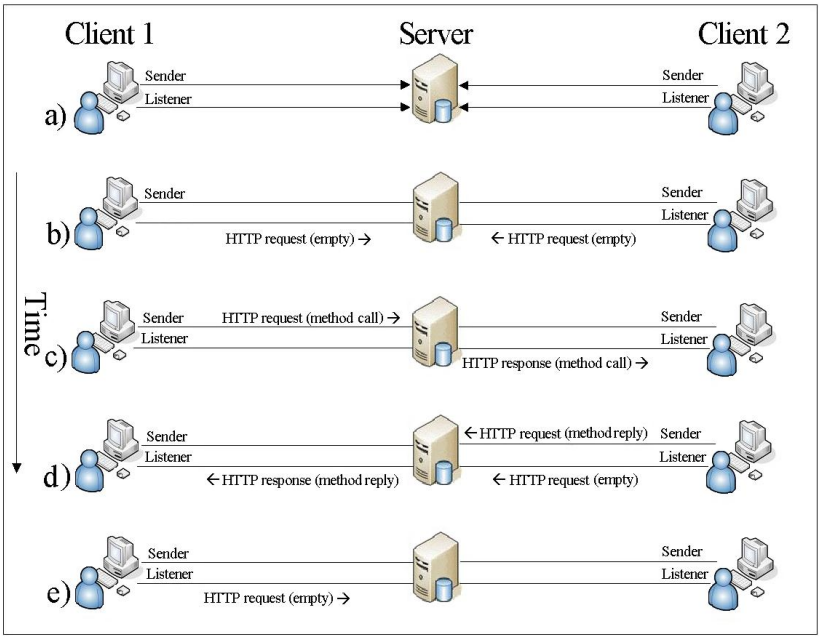

Figure 9: Set up of a two-way HTTP(S)-channel for client-to-client method invocation. The connection is initialized by two clients (Justervesenet and customer) each setting up a two-way HTTP(S)-channel to the server

HTTPS-channel in the system, basically by implementing the SSL/TLS protocol.

In more detail the connection is made using the property of a HTTP-connection [51] which is kept alive during a session. The establishment of this connection is illustrated in more detail in Figure 8. Each client sets up keep-alive HTTP connections to the server, *sender* and *listener*. The reason why two connections are needed is the asymmetrical request-respond design of the HTTP-protocol. The *sender* connection is later used by the client and the *listener* connection is used by the server. Since the solution is based on reuse of HTTP connections set up by the clients, the solution will work with any firewall or proxy server. The client will always have an 'unanswered' or pending HTTP request on the *listener* connection which the server may use to send data or method calls in an HTTP response.

Initially, a client sets up two HTTP-connections to the server, which are named *sender* and *listener* in the figure. The *listener* connection is in a constant pending HTTP request mode. While these two HTTP-connections are still open, the client uses the *sender* HTTP-connection to send data or method calls to the server, and the server might use the *listener* HTTP-connection to send data or method calls to the client. Typically, the timeout of the pending HTTP request is about 1-2 minutes. If the server does not send any data before timeout, the request is cancelled and a new request is initiated. When the client-to-server method and the server-to-client method are combined, two clients may call methods directly at each other. This is illustrated in Figure 9.

4.2.2 Instrument Operation

For instrument control, the system uses the Virtual Instrument Software Architecture (VISA) [52]. VISA is a standardized interface between the control applications and the instruments. A pre-installed DLL-file(dynamic link library), the visa32.dll, supports the control interface with several methods for hardware (i.e. instruments) communication, for the iMet v1, this is limited to GPIB [53] support. A .NET component, called the GPIB Communicator, is implemented to load the VISA library at start-up. Four methods are included in this component, *FindResources*, *Read*, *Write* and *Query*. An overview of this is shown in Figure 10.

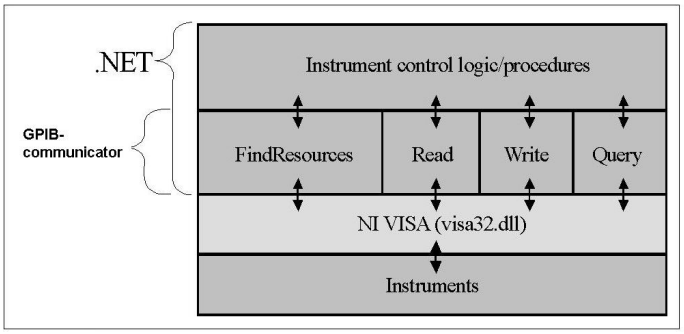

Figure 10: The .NET component, GPIB Communicator, supports procedures calls with four methods. These methods could then be used to access instruments via the standardized VISA interface

By using the GPIB Communicator, all measurement setup and control is done in the .NET environment. The instruments connected to a customer's computer are made available to remote authorities by exporting the four methods, which handle the hardware communication. The *Virtual Instrument Repository* is a set of software representations of

all connected instruments. This repository contains instrument-specific commands and state-information for each instrument. To access individual instruments, an instrument identification string is used together with the appropriate method, as shown in Figure 11.

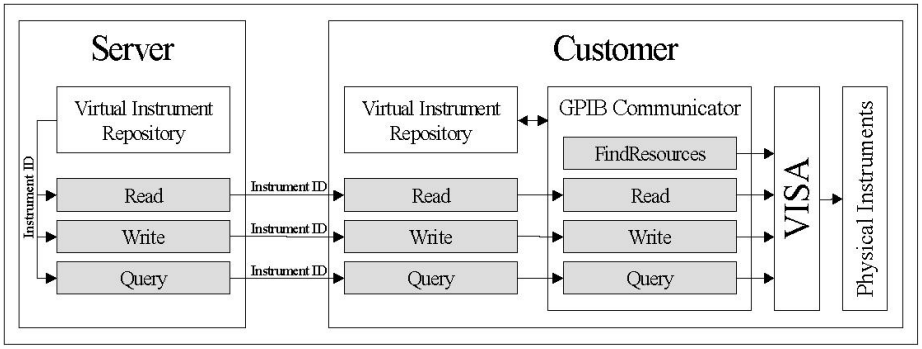

Figure 11: The server only uses the customer's Read, Write and Query methods to communicate with the instruments located at the Customer. An instrument identification string is used as input to these methods in order to access the correct instrument

An overview of the iMet system v1 communication architecture is shown in Figure 12. Here is shown the interconnection between the customer, server and authority computers, and also the virtual path from the authority and server to the customer instruments, indicating that the calls made at the authority or server computer virtually controls the instruments at the customer's site.

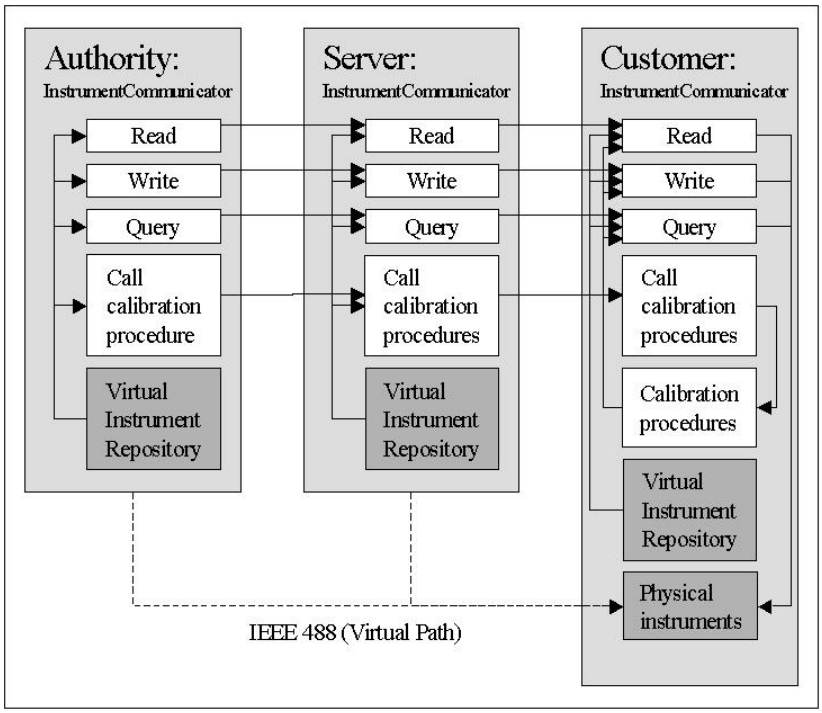

Figure 12: The iMet system v1 communication architecture. Implementation of the InstrumentCommunicator interface enables the Authority application to operate a Customer instrument using the Read, Write and Query methods or run measurement procedures

When a customer connects to the server, the customer's instrument bus is scanned and for each instrument found, an instrument software object is added to the virtual instrument repository at the customer and at the server and also at the authority if one is connected.

Measurement and calibration procedures, pre-installed at the instrument client, uses the *Read*, *Write* and *Query* methods to communicate with the physical instrument. This InstrumentCommunicator interface is implemented at all applications - authority, server and customer. The instrumentID is used to to access specific instruments, and in this way, each call of a method will send the command to the correct instrument. Calibration procedures may be run on any host implementing the InstrumentCommunicator interface.

4.3 iMet v2

In the second version of the system, the emphasis is mainly on building scalability into the system, i.e. the possibility to add new measurement procedures (software). The implemented communication architecture is shown in Figure 13. By using this architecture, an application may operate a customer's instrument using *Read*, *Write* and *Query* methods or run calibration procedures. The calibration procedures could be downloaded from a database and run during runtime, without system restart or manually recompilation.

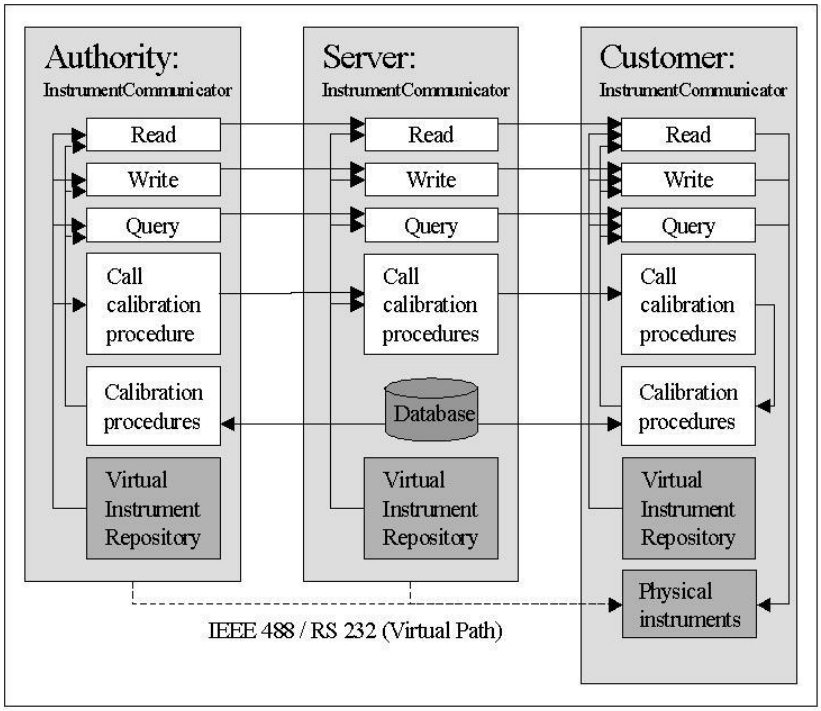

Figure 13: The iMet system v2 communication architecture. Implementation of the InstrumentCommunicator interface enables the Authority application to operate a Customer instrument using the Read, Write and Query methods or run measurement procedures. The measurement procedures may be downloaded from the database at the server and run during application runtime, without system restart or manual recompilation

The C# programming language has the capability of compiling source code in memory during runtime. A running C# object may take a piece of source code, describing

another C# object and compile and instantiate it. In this way the new object is incorporated in the running object, and it may be used as if it was compiled beforehand. This feature is used in the iMet system version 2 (and also in version 3), to make it possible to download and dynamically compile and run measurement procedures available as source code. The customer application now includes functionality to download and compile measurement procedure source code and also a container for running the compiled procedures. This is shown in Figure 14.

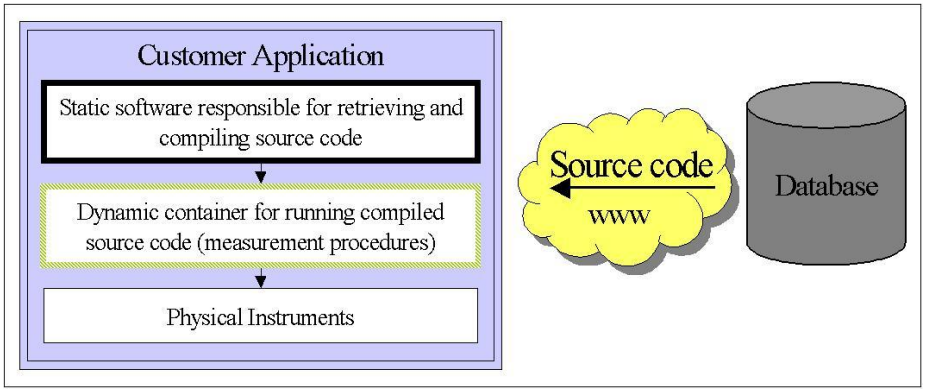

Figure 14: The iMet system v2 customer application architecture

4.4 iMet v3, a Generic Instrumentation System

The latest version of the system is a result of a joint project between Justervesenet and NPL, with the aim to create a highly adaptable instrumentation system. The expressed vision for the system is that it should support remote operation of potentially any device from anywhere using any measurement procedure.

To obtain this, a second property of scalability is added to the system, by including the possibility to dynamically add drivers for new instruments to the system. The motivation for adding this property is described in [13]. Today, instrument-controlling software and hardware drivers are often manufacturer-specific and custom made. To take this into account, and possibly make use of this variety of non-standardized instrumentation controlling software, flexibility in adding drivers for new instruments to the system will be of great importance.

The version 3 of the iMet system facilitates to adapt to this flexibility requirement by introducing a somewhat different architecture compared to the earlier versions. The description of the system now also attempts to include this generic and flexible property by using more generalizing terms. The system is presented as a framework for PC-based instrument control. It is now divided into four parts as shown in Figure 15.

As seen in the figure, the use of the terms *server* and *client*, now has a different approach compared to how the terms were used when describing the two earlier versions. The server now denotes the computer that is the server for the devices, earlier denoted as the client or customer, and the client term is used for the operator, earlier described as the authority's computer. In addition the server term is also used related to the database server and the communication server (which was the same in earlier versions).

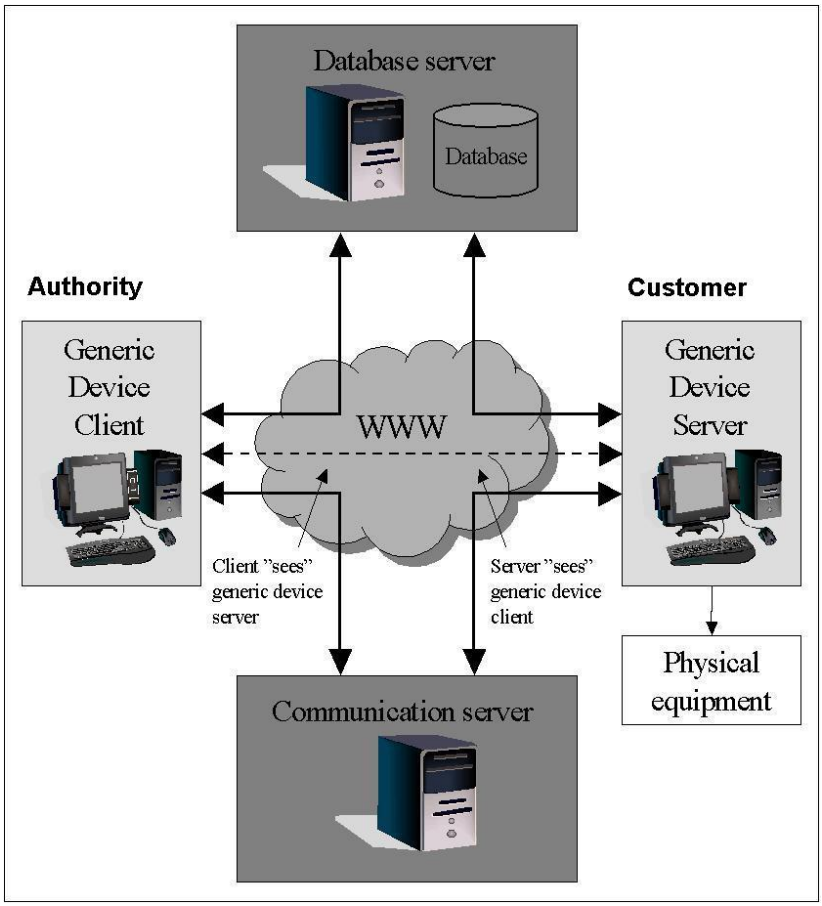

Figure 15: iMet version 3, generic system architecture

1. A **generic device server** application (GDS), responsible for communication with the instruments and providing services to control them.
2. A **generic device client** application (GDC), acting as the device operator.
3. A **database server**, containing all instrument-related information and measurement procedures.
4. A **relay communication server**, handling the GDS-GDC communication.

The main application is run on both the operator (in earlier versions denoted as 'Authority') and customer computer, and contains methods to instantiate both GDS and GDC objects. Only one GDS per computer is allowed, while there are no limitations to how many GDCs can be run on a single computer, since these are supposed to operate remote GDSs. It is possible for a GDC to operate a GDS on the same computer; in this case the communication is set up between processes. But for the purpose of this thesis we will look into the scenarios where a GDC operates a remote GDS, when the communication uses TCP/IP.

The GDS is shown in Figure 16. It comprises a *service manager*, which is responsible for all client connections, and a *control manager*, which is responsible for handling the device (instrument) communication. The control manager can communicate with a de-

vice directly, using a specific device object, or it can communicate with a device indirectly via measurement objects. When the GDS sets up the communication to a instrument, it first has to download the instrument-specific information from the remote database. The information is downloaded in the form of source code, and this source code is compiled and instantiated at runtime. In this way, the GDS needs no prior knowledge of the devices it communicates with, and the remote database is effectively a part of the instrument control application, enabling a dynamic system, which can adapt to a changing environment and various instrument setup procedures.

The server communicates with the devices through an interface, the GDS interface, which presents several methods. The most important methods are:

- **activateInstrument(...)** Activates and registers a specified instrument. This includes the downloading of instrument information, compiling of the source code and starting a service to communicate with the instrument.
- **deActivateInstrument(...)** Deactivates and unregisters a specified instrument. This includes closing down the instrument connection and removing the service to communicate with the instrument. The compiled code is actually not removed, so the instrument could be reloaded later without recompilation of the source code.
- **run Measurement(...)** Compiles and runs a specified measurement procedure. The source code for this measurement procedure is received from the calling operator, together with necessary measurement configuration information.

The GDC is shown in Figure 17. The GDC may control remote devices directly by using runtime-compiled graphical user interface (GUI) components for specified instruments, or indirectly using runtime-compiled measurement objects. As seen in the figure, the client can communicate with a GDS using different channels, e.g. Internet, WAN or LAN, or it can communicate with a local GDS (on the same computer).

The *communication server* shown in Figure 15 has the same function and operates in the same way as the described server in the first version of system. The *InstrumentCommunicator*, both used in version 1 and version 2, is still used as well as the *Read*, *Write* and *Query* methods. In version 2 the server also included the database for measurement procedures. This functionality is now located in the *Database server*.

4.4.1 Instrument operation iMet v3

Prior to any instrument operation the following are sent to the customer:

- A calibration standard (measurement instrument)
- A CD containing the IMet application and the customer x.509 certificate (with the customers private key), issued and signed by Justervesenet and username/password for logon to the server when the application is started. The content of this CD is encrypted.
- A password for decryption of contents of the CD, is distributed in a letter or by telephone to known customers.

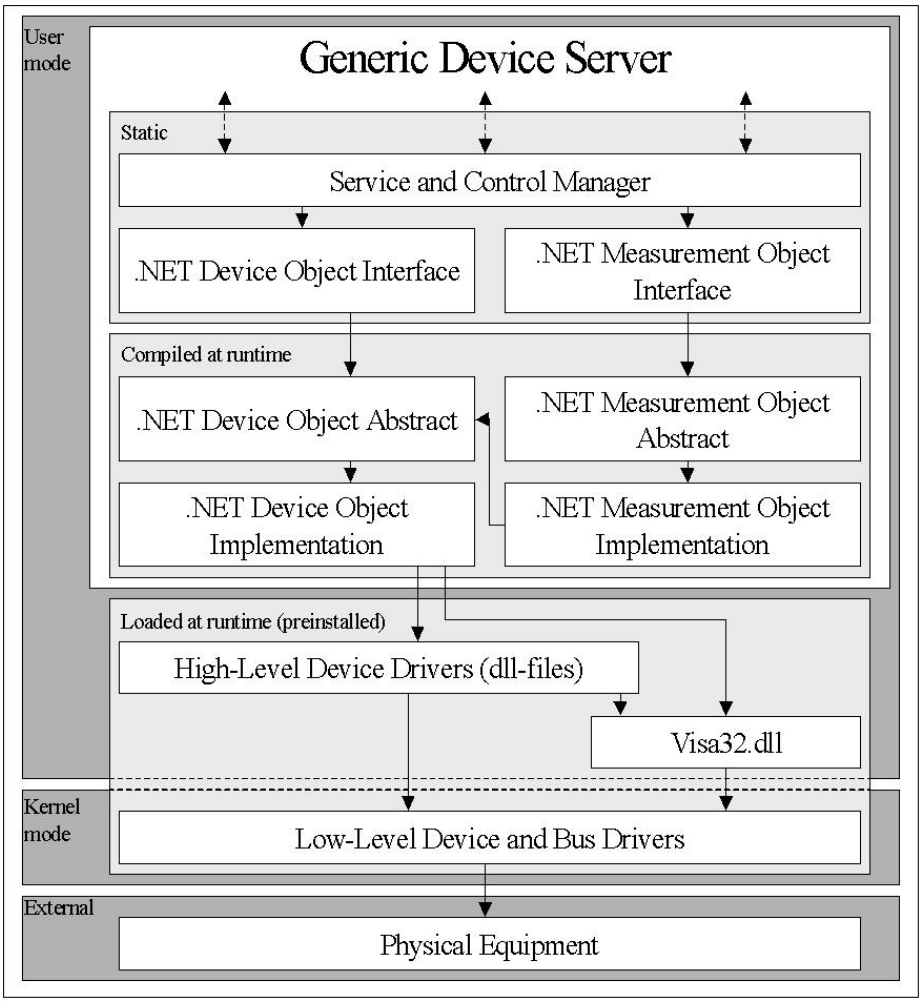

Figure 16: The Generic Device Server Architecture. The figure shows the double functionality of the GDS, it is possible for the server to communicate directly with the instrument, via the .NET Device Object Interface, or indirectly via the .NET Measurement Object Interface.

Before being able to operate an instrument connected to a remote computer the following will have to be performed on the remote computer (The instrument itself should also be connected, together with standards etc):

- The native hardware bus driver should be installed
- The instrument driver and API should be installed
- Some manual work might be necessary to do (simple shell commands) by a local operator
- A local operator should start the GDS application and connect it to the (public) communication server

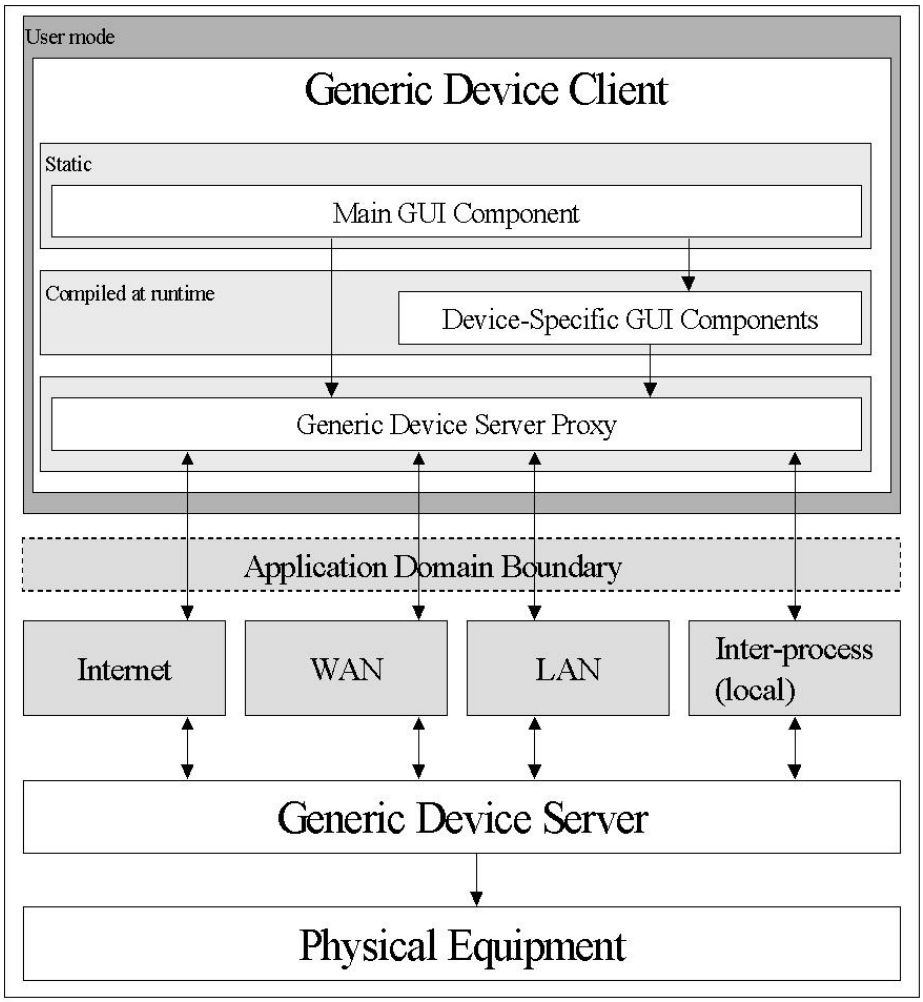

Figure 17: The Generic Device Client Architecture

When the local operator has performed the initial procedure manually, the following is performed by the main application:

- The GDC downloads an instrument-specific GUI component from the database server.
- The GUI component is compiled and instantiated (runtime).
- The GUI component contains controls to operate the instrument directly. When a function is chosen, the selected control signal is serialized into a byte-stream and sent to the remote GDS via the communication server (if remote operation...
- The GDS de-serializes the byte stream and sends the method call to the correct instrument software wrapper.
- The software wrapper converts the method call into a suitable instrument driver call.

Measurement results are returned from the server (customer computer) in much the same way.

For each instrument there are two software components; a GDC-side GUI control component and a GDS-side high-level instrument driver component. The details of the communication with the instrument driver component are handled by the GUI control component, which is compiled and instantiated during runtime.

4.5 Security Measures Implemented in the System

A system like the iMet, which involves different parties in the operational procedures and that is built to operate using the Internet as the communication channel, will clearly have to deal with several information security issues.

For the purpose of this thesis, we will not go into all the technical details regarding the implementation of the application and the security services, this is due to different concerns both regarding disclosure of technical information and also due to the depth of the analysis that is performed. We will come back to and define limitations for the work and assumptions made in section 4.6. When describing the security measures implemented in the system, we will therefore mainly focus on the functionality of the different security measures rather than going into elaboration of all the technical details.

In the design of the system security issues were addressed and several security measures were built into the system. We first introduce the basics of the mechanisms in sections 4.5.1- 4.5.4. Sections 4.5.5 an onward describes then how these mechanisms relates to the different areas of the application system and affects the parties involved.

4.5.1 Implementation of a HTTPS-Channel

To deal with issues of confidentiality and integrity, the system has implemented mechanisms for encryption and digital signing of data that is sent in the communication channel. The system uses x.509 certificates [54], and private/public key pairs for the parties involved in the communication process.

All communication between the parties uses the HTTPS-protocol (HTTP, [51] over SSL/TLS, [55]) in the full-duplex channel that is set up by the system. This basics of the full-duplex HTTP-channel was described in section 4.2.1. Figure 9 on page 22 shows how the two-way HTTP channel is instantiated, but this will be similar for a HTTPS-channel. The only differences will be that that port 443(HTTPS) instead of 80(HTTP) is used on the computers involved, and that the SSL/TLS protocol is used, including the SSL/TLS Handshake protocol in the connection setup (see Figure 18).

Authentication of both server and client is performed when the connections are set up, by using the SSL/TLS Handshake protocol shown in Figure 18. The application is built to check whether the certificates that are sent in the handshake are correct and trusted and thereby the correct entities are involved. The server x.509 certificate (which has been sent to the customer from Justervesenet together with the transport standard) is used in the application to implement the server as a trusted Certificate Authority at the client's computer. When the SSL/TLS handshake is then initiated, the certificate sent from the server will then be a trusted one. The server's IP-address is also sent to the customer on the CD together with the application. The application at the customer's computer connects to the server identified by this IP-address, and checks whether the certificate sent in the handshake is a trusted one, i.e. issued by Justervesenet's server. If this is not true, the application will not be able to run.

Customers will prior to engaging in the Internet-enabled calibration process have to

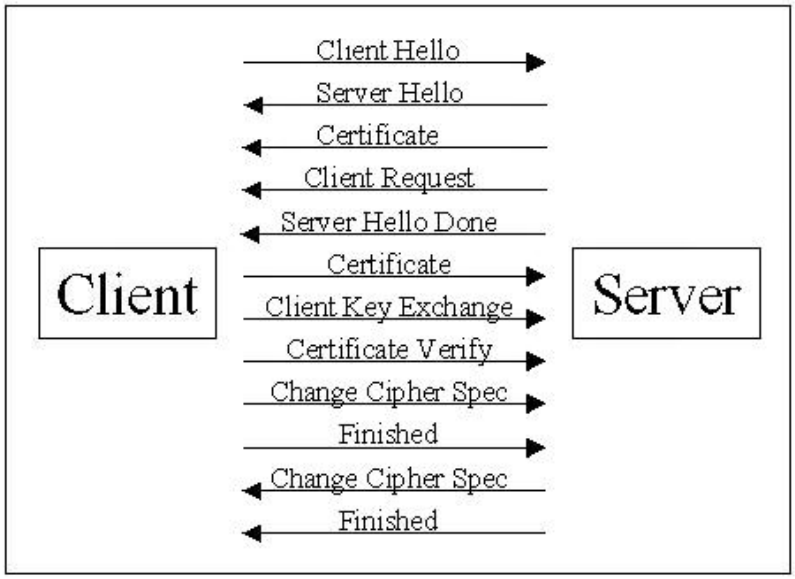

Figure 18: The SSL/TLS Handshake protocol, with server and client authentication. During the initialization, the client and server agree upon encryption algorithms, perform certificate-based authentication and create shared symmetric key for the SSL/TLS session

request Justervesenet for a server-signed x.509 certificates, and this certificate is used to authenticate the customer's computer when connection to the server is established.

Justervesenet as the authority in the calibration procedure will have issued server-signed x.509 certificates, and this certificate is used to authenticate the computer of the operator at Justervesenet when connection to the server is established.

The latest versions of the SSL/TLS-protocol has proved to have a high level of security for the security services it is designed to handle, as shown in [56] and [57]. It is however important to mention that the SSL/TLS protocol only ensures security on the transport layer (in the OSI reference model [58]) in the communication channel that is set up. Security at levels higher up in the OSI-model, and thereby application layer security, will still depend on how the protocol is implemented in the application.

4.5.2 Source Code Integrity

In version 3 of the system, the possibility to dynamically add drivers for new instruments to the system is introduced. To make this possible, program source code has to be downloaded from the database at the server, compiled and run at the customer's computer while the application is running. It will be vital to secure integrity of code to be run on the customer's computer. Malicious code could be inserted in the database or in the communication channel by i.e. a hacker and result in severe damage at the customer's site when the code is compiled and run. To mitigate this, the source code is signed by the server's private key in the database, and the application verifies this signature by using the server's public key in the servers x.509 certificate, before the code is actually run at the customer.

4.5.3 Distribution of Security-Related Credentials

Distribution of credentials related to security issues is described to some extent by Sand in [13]. Here we describe a procedure for distribution of the security related credentials that goes further than this description, and which has not yet been fully implemented, but we assume for this thesis work that the intentions of the described procedure will be followed.

Any customer in an Internet-enabled calibration process will have to get a server-signed x.509 certificate from Justervesenet. A CD is sent to the customer together with the calibration standard, or in a separate sending. This CD encompasses the main application software (included the server IP-address), the customers x.509 certificate signed by Justervesenet's server and username and password for logon to Justervesenet's server when the application starts up. The content of the CD will be encrypted and the password necessary to decrypt the contents and start the application will be distributed by telephone or in a separate sending to the customer.

4.5.4 Procedures for Distribution and Trust

The need for procedures related to the handling of credentials like username/password cryptographic keys and certificates is addressed in [13], but not clearly implemented in the system, this is mainly due to the fact that the system is still in a prototype phase and not implemented in a commercial service. Justervesenet issues the x.509 certificates and the related keys, at the time being by using the management application located at the computer that also hosts the database server and communication server.

In the future, the certificates to be involved in the connection set up and authentication procedure are intended to be signed by a trusted certificate authority. This functionality is not implemented yet, so the customer would have to trust Justervesenet for the security in issuing, distribution and handling of certificates and key-pairs.

4.5.5 Security in the Communication Channel

The security mechanisms built into the system are initially mainly focused on securing the data in the communication channel between customer, authority and the database and communication server.

Confidentiality

Confidentiality is obtained for data sent in the communication channel between the nodes in the system, by implementing the HTTPS-protocol for all communication purposes. During the client-server session all communication signals are encrypted according to the SSL/TLS standard. Initialising a SSL/TLS session includes several steps;

1. Client and server negotiate the encryption algorithm to be used
2. Authentication of the parties are performed based on public/private keys
3. A shared key for the session is agreed upon, this includes key length
4. Symmetric encryption/decryption is used for the session

The confidentiality is thereby secured between the two parties communicating, but it will depend on the authentication procedure to ensure that the correct entities are involved.

Possible threats to confidentiality in the communication channel could be imposed by vulnerabilities if algorithms/keys that are negotiated/selected are not strong enough and thereby making it possible to perform attacks like man-in-the-middle or replay. (Should need procedures to ensure that old versions of SSL with known vulnerability are not used)

Integrity

The underlying SSL/TLS protocol is also used to ensure data integrity in the communication channel. Integrity is achieved by the signing of all data exchanged by the client and server in the communication process, using the private key of the sender to sign and the public key to open at the receiver's side. SSL/TLS only ensures the integrity of the individual data packets that are sent are secured between the two parties communicating. Integrity of application level data content, such as measurement results and program source code is still not secured, and here also applies that it will depend on overlying application and the authentication procedure to ensure that the correct entities are involved, and that the integrity of data on application level is secured.

Authentication and access control

SSL/TLS could be used to authenticate specific entities, such as companies, specific computers and specific user accounts on a computer, and these mechanisms are used in the iMet system both for client and server authentication. On Windows systems, the private key (associated with a x.509 certificate) that is used for authentication is normally stored encrypted using a key witch is generated based on the logged-on users password. In the iMet system, access control based on iMet username and iMet password for logon to the application is also implemented.

Availability

Availability of the iMet system will to a great extent be dependent of a reliable communication channel. This has been addressed by implementation of the bi-directional communication channel described in in section 4.2.1. The properties of this channel are developed to ensure reliable communication between two entities operating behind firewalls and proxy servers on different LANs. The availability of the communication channel of the system will still to a great extent be dependent on the Internet and different components in the communication channel for timing issues.

Non-repudiation

Non-repudiation will be a service that should be addressed when the sender and receiver in data transmission needs proof of the origin and delivery of data. There are not implemented any specific security measures to ensure non-repudiation in the iMet system. This will have to be implemented on application level by using digital signing mechanisms. Potential data to be signed would be data concerning measurement results and measurement configuration.

4.5.6 Security at Customer Site

Prior to engaging in any Internet-enabled calibration procedure, it is expected that customers should undergo adequate user training and receive system documentation describing how the system should be used.

The customer's security policy and how this is maintained, including the implemented up to date fire-walls and virus protection, will be of vital importance to maintain the

security level of the system application Justervesenet will have to trust the customer to deal with these issues in an adequately secure way.

The security of systems like the iMet system will to some extent have to be based on trust between the involved parties. To some degree the level of this trust could be heightened by the use of regulatory contracts expressing responsibilities and judicial obligations for the parties.

Confidentiality

At the customers site it will be vital to ensure that the private key, the iMet username and password and the password used to decrypt the application CD are used and stored in a way that maintains the confidentiality of these credentials. Confidentiality will be dependent on how the iMet system credentials are managed, including the implementation of access control to the customer's computer.

Confidentiality of measurement results and other information about the customer's instruments and system will be secured in the communication channel by encryption/decryption mechanism in the SSL/TLS protocol, but the security in this mechanism relies on secure handling of the essential credentials for the communication.

Integrity

It will be of vital importance to secure integrity of metrological parameters at the customer site, i.e. measurement result from calibration, information about the instruments in the setup, measurement procedures and other environmental parameters. The integrity will be dependent on the implementation of the system application, integrity of downloaded code is checked before it is compiled and run at the customers computer, but there is not implemented any aimed measures to eliminate the possibility for a skilled fraudulent user to insert code that could be run at the customer site and simulate or manipulate measurement results.

Authentication and access control

In addition to authenticating persons and computers as described earlier, each connected instrument at the customer site should also be authenticated. This authentication is for the time being performed in the system application procedures by checking the instrument ID information stored in the instruments memory.

Non-repudiation

The need for non-repudiation mechanisms is not addressed so far in the system development. What could be necessary to secure is that the customer not would be able to delete or tamper with e.g. 'unwanted' data or measurement results. As the calibration service is a commercial service it should also be secured that the customer could not deny that the service has been executed, and therefore refuse to pay for the service. This will be taken care of by the fact that the customer will need a calibration certificate to employ the calibration results. This calibration certificate will be issued after the calibration is finished and billing will be based on the issuing of this certificate.

Availability

Besides the implementation of the communication channel, there has not been implemented any special mechanisms to secure availability of different components in the system application. The availability property includes reliable communication, correct timing and availability of the correct versions of software components in the system. It

should be said that the characteristics of the application does not claim for any special requirements regarding availability, the instruments are normally operating on the relatively slow GPIB-protocol, though not very time critical, and the measurements could be restarted if there is a hang up in the system. Possible attacks to the system could be DoS-attacks.

Security policies

For the security mechanism described here, the security level will necessarily depend on implemented policies, both at customer and authority side, on how to handle certificates and keys, usernames and passwords etc. Negotiation of strong algorithms and appropriate key lengths are vital to obtain a satisfying security level, together with implemented up to date firewalls and virus protection.

The responsibilities of the customer should be defined and regulated by a contract that is to be signed prior to the security related credentials are distributed and calibration is performed.

Before using the iMet system application, customers should receive adequate user training and documentation for the system.

Physical access

To support the requirements for confidentiality and integrity of the system, physical access to the customer's computer should be restricted. The limitation of physical access will have to be maintained according to the customer's security policies. Normally customers already have restricted access to the locations where the instruments are kept, out of the consideration to protect the instruments from theft or other unwanted actions.

4.5.7 Security at Authority Site

At the authority site (Justervesenet in this case) much the same considerations as for the customers will apply regarding security issues.

The operator at Justervesenet should undergo adequate user training and have system documentation describing how the system should be used.

Justervesenet's security policy and how this is maintained will of course also be of vital importance to maintain the security level of the system application. Here the customer will have to trust the Justervesenet to deal with these issues in an adequately secure way.

A contract between the parties will of course also regulate and present judicial responsibilities for Justervesenet.

Confidentiality

At the authority site it will also be vital to ensure that the private key and the iMet username and password are used and stored in a way that maintains the confidentiality of these credentials. This also applies to other information about the iMet system. Confidentiality will be dependent on how the iMet system information credentials are managed, including the implementation of access control to the operator's computer at Justervesenet.

Confidentiality of measurement results and other information about the customer and the customers instruments and system will be secured in the communication channel by encryption/decryption mechanism in the SSL/TLS protocol, but the security in this mechanism relies on secure handling of the essential credentials for the communication at Justervesenet as well.

Integrity

Justervesenet should have a security policy and necessary procedures to ensure integrity of metrological data like measurement results, information about calibration standards and measurement results in calibration certificates.

Authentication and access control

At described earlier the computer of the operator at Justervesenet is authenticated to the server by use of the server-signed x.509 certificate when connection to the server is established. In addition username and password for the authority operator is used to restrict access to the server.

Non-repudiation

As mentioned earlier non-repudiation mechanisms are not addressed so far in the system development.

Availability

To be able to run the system application the operator at Justervesenet and necessary and correct software and hardware components of the system should be available and have available sufficient resources. No specific measures have been considered to ensure this, as this will normally be part of administrative procedures and prioritizing.

Security policies

For the security mechanism described here, the security level will necessarily depend on implemented policies, also at authority side, on how to handle certificates and keys, usernames and passwords etc. Negotiation of strong algorithms and appropriate key lengths are vital to obtain a satisfying security level.

Physical access

Physical access to the operators computer will have no special restrictions other than normal access restrictions as locked doors outside normal working hours.

4.5.8 Security at Database Server

Security regarding the database server will be of vital importance for the iMet system. Both communication server application and the database server application, containing all device-related information and measurement procedures are located at the same computer. For the time being also the management application and development applications are also located here.

Vulnerabilities in the database management system (DBMS) or the implementation of the database and server will possibly have serious impact on the security level of the system application.

Here we will not go into full detail about all the technical details regarding this. This is both due to considerations regarding disclosure of sensitive information for Justervesenet and due to the fact that the system is still in a development phase and much of the details regarding the implementation of the databases and management of these are supposed to change in an operating commercial system.

Integrity

The underlying SSL/TLS protocol is used to ensure data integrity in the communication channel between customer and server and between authority and server. Integrity of source code in the data base server is partly secured by the use of digital signature using

the server-issued private key. The integrity of both digitally signed source code and other sensitive information about measurement procedures, measurement results, customers and authority, keys and certificates as well as information about the iMet system will also depend on routines for access control to the server computer and applications as well as back-up and maintenance routines.

Confidentiality

Sensitive information as measurement results, information about customers and authority, private keys for digital signatures and authentication purposes as well as information about the iMet system will be stored in the computer hosting the database server and communication server. Confidentiality of this information will depend on routines for access control to the server computer and the different applications here as well as implemented routines regarding this.

Authentication and access control

At described earlier the computer of the operators at Justervesenet and customer is authenticated to the server by use of server-signed x.509 certificates when connections to the server is established. In addition usernames and passwords for the authority and customer operators are used to restrict access to the server. The access to the system is role-base and both the customer and authority will have restricted access to the system. For a customer it will for example only be possible to communicate with his own instruments. The authority will have less restriction, but will still be a user of the system with no access for maintenance or making changes to the system. Access to the management application for the system will be limited to users at Justervesenet based on username and password.

Non-repudiation

As mentioned earlier non-repudiation mechanisms are not addressed so far in the system development.

Availability

Standard backup routines are implemented for ensuring availability of the correct software components. At the moment, availability issues have not been addressed to a great extent, e.g. there is not implemented any redundancy for hardware components. For a commercial system it is expected that these issues will have to be considered more closely.

Security policies and procedures for management and maintenance

As for the customer and authority sites, the security level for the system will necessarily depend on implemented policies and procedures for this. These policies should express requirements regarding the handling of certificates and keys, usernames and passwords etc., as well as requirements regarding the use of strong algorithms and appropriate key lengths. Issues regarding management and maintenance of system-related software and hardware should also be addressed, including definition of responsibilities.

Physical access

The server (database and communication) is placed in a locked server room, with physical access restricted by keycard with pincode.

4.6 Assumptions and Limitations for a Security Analysis

The iMet system has been under development and in prototype phase for quite a long time. It has still not been implemented in a commercial service. The development is still not finished, and for the purpose of this thesis it is necessary to make some assumption regarding the environment where the system is to be used, as well as for the actual version of the system being analyzed.

Therefore, it has to be stated that the actual system being analyzed in this thesis, will be the system described here, with all details as given in the description in the preceding sections. If any improvements or other changes have been made during the time of writing this thesis, their implications have not been considered here.

The customer's system setup is not known in detail. For the purpose of this analysis work we will have to make some generalizing assumptions when it comes to the details about this. The same will also be necessary when it comes to Justervesenet's IT systems, including LAN/WAN components and setup. As mentioned in the Introduction, chapter 1, security concerns regarding confidentiality of system specific details and information, restrict the degree of details in the information that are being exposed, and what types of information that actually are to be exposed. Figure 19 shows an schematic overview of components, including the iMet database server and communication server.

The database server and communication server described in version 3 is for the moment implemented as two different applications on the same computer. This computer also holds the administrative application used to maintain the system. It is expected that changes will be made to this configuration prior to putting the system into commercial use. We have still chosen to perform the analysis based on how the system is implemented at the time being. It would have been difficult to perform a sound analysis of the system trying to define a configuration that is very different from the one actually implemented.

XML-blaster was introduced as the middleware used in version 3 of the system. Later an implementation using .NET Remoting was developed. For the purpose of this thesis we will describe the system using the latter implementation.

4.7 Components in the System

To summarize the system description in the preceding sections we here give an overview and a brief description of the different components in the system.

4.7.1 Hardware

The topology of Justervenenet's LAN could be described as a standard LAN topology, a schematic of the main components including components used for communication on the Internet is shown in Figure 19.

Authority computer

This is one of the computers at Justervesenet, normally located in Justervesenet's building, on Justervesenet's LAN.

Customer computer

This would be a standard PC, we do not know much about this, but it would be expected to be a normally equipped PC. We also suppose that the PC will be maintained due to a company security policy, e.g. safeguarded by installation of standard virus protection,

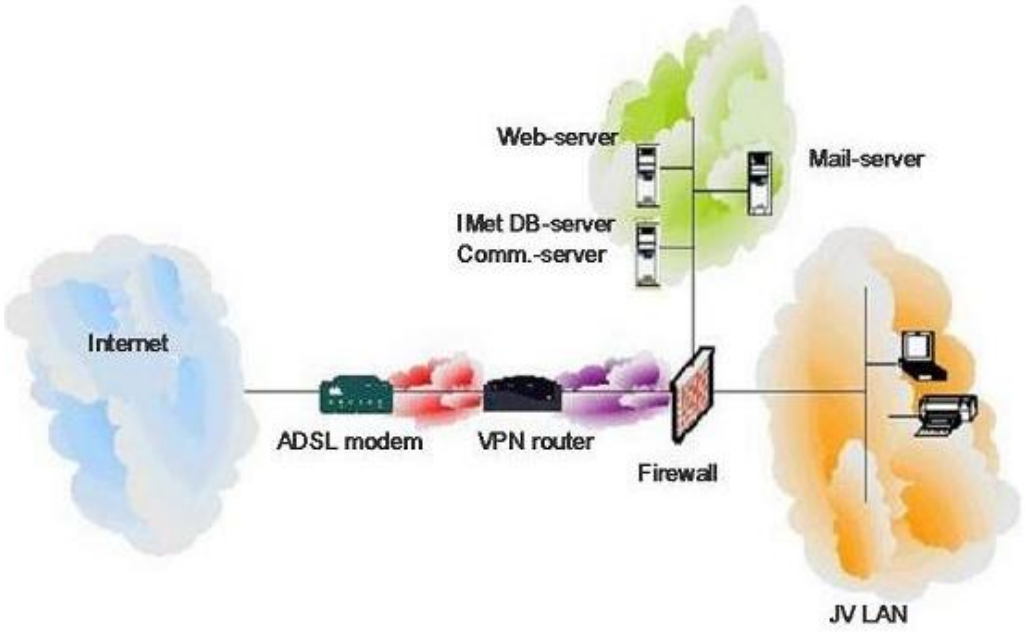

Figure 19: A scematic of Justervesenet's LAN and Internet connection topology

a protective firewall with a normal level security configuration and in a program with regularly implementation of security updates for operating system and other software components running on the system.

Communication server

This is a standard PC running the communication server application.

Database server

This is a standard PC running the database server application. For the time being both the communication server application and the database server application are hosted at the same computer.

WAN/LAN components

These hardware components includes different switches and other network components on both the customer's and Justervesenet's networks.

4.7.2 Software

The software in the iMet system could be divided into several components as follows;

- Main application
  The main application will run on both the operator and authority computers.
- Two-way communication channel
  The communication channel uses the GenuineChannels design of a full duplex HTTPS-channel, and is described in section 4.2.1

- GDS
  The Generic Device Server is described in section 4.4
- GDC
  The Generic Device Client is described in section 4.4
- The database server application is installed at the database server computer
- The administrative application is used for further development and management of the iMet application
- Measurement procedures are available from the database server as source code. When initiating a calibration of an instrument, the corresponding source code is downloaded, dynamically compiled and run at the customer.
- Instrument drivers are implemented as high-level device drivers and are included in the calibration procedures.
- Instrument Bus drivers i.e. VISA or specific hardware bus drivers (e.g. GPIB) are implemented in the high-level device drivers

4.8 Assets in the System, Attack Goals

The iMet system could be defined as

> *the hardware and software components dedicated to perform the necessary activities in order to initialize and perform instrument control and data transfer, and to fulfil the requirements specified by the measurement procedures chosen by the operator of the system.*

In this section we present the assets of the system. This will be the various system components of both customers and Justervesenet, and the asset concept includes the qualities and characteristics of these components, which also should be protected. In this section we mainly identify the assets and in the following chapter 5, we will come back to an evaluation of criticality of the assets. For further analysis we will select some of the assets based on this criticality evaluation. It should be mentioned that the list presented here is not a complete one, we have selected some of the most important assets to be a basis for later analysis:

4.8.1 Measurement Related Data at Customer/Justervesenet's (Authority)

For the validity of the calibration process it will be important to protect the data resulting from and related to the calibration procedure. This will apply for several stages of the calibration process;

- In Database
- At customer
- In the communication channel (on the Internet)

Measurement data includes the records of individual measurement results, registered parameter settings, timestamps for each measurement, calibration standard data used as input to the measurement as correction factors and identification information of test object and calibration standards. For all these measurement data related assets, the following properties are to be protected:

- Integrity (correct measurement results, correct calibration data of standards)

- Confidentiality (customers measurement data)
- Availability (loss of measurement results data, calibration data of standards)

4.8.2 Calibration certificate

The measurement results and data related to these are the basis for the final values in an issued calibration certificate. The processing of data from measurement results to values in a calibration certificate should also be protected.

- Integrity
- Confidentiality
- Availability (archive functionality)

The calibration certificates are for the time being issued semi-manually. Measurement calibration values are controlled both by the operator of the system and the person that signs the certificate. This control includes evaluation of results due to measurement history, standard deviation and experience with this specific calibration object, or similar objects. A later version of the system is expected to implement an automatic issuing of calibration certificates, and the control could to some extent be built into the system.

The measurement data used as input to the calibration should be kept available together with a copy of each calibration certificate for a minimum of 6 years, due to Justervesenet's policy.

4.8.3 Customer/Justervesenet's (Authority) Instruments

Several aspects of protection related to the instruments used in the calibration process should be addressed;

- Correct set-up of instruments
- Correct measurement procedures
- The physical instruments, prevention of damage
- Correct identity of calibration standard and measurement objects (instrument)

The instruments in the setup are comprised of both the customer's instrument(s), the test object in the setup, and Justervesenet's calibration standard instrument.

4.8.4 iMet System Assets

It will be of great importance to ensure the functionality and protect the customer's and Justervesenet's interest, by protecting the different components of the iMet system;

- Hardware system components, that is server computer
- Software system components, and the security attributes related to these that is integrity, confidentiality and availability. The software system includes both:
  - iMet Communication System
  - Measurement procedures and instrument drivers
- Information about the system (which could make other attacks possible), this includes:
  - Certificates and cryptographic keys

- Usernames and passwords
- Information about configuration

4.8.5 LAN at Customer and Justervesenet's (Authority)

The network systems and relating assets at both customer's and Justervesenet's sites should also be protected, these are:

- System components - hardware
- System components - software (integrity, confidentiality, availability)
- Data regarding the measurements and calibrations
- Information about the system that for example could make other attacks possible

The LAN at customer and Justervesenet's site includes hardware; PC's, servers, switches, printers, modems etc., and software that run on these hardware components as well as configuration and settings of these. A part of the LAN with respect to a definition of possible attack goals will also be procedures that relate to Justervesenet's security policy, general maintenance and monitoring of activities.

5 Applying the Attack Tree Method

As discussed earlier, the attack tree method provides a structural tool for discovering possible vulnerabilities in a system, and the methodology could be used as part of a risk analysis process. The implementation of the method could be made both for identification of security problems at design and development phases as well as for concerns regarding reliable operation of a system.

It will also be possible to perform the analysis for a system or sub-system in specifically defined contexts and environments, as long as the outline and interfaces are defined. This makes it feasible to split an expressed set of attack paths leading to a high-level attack goal into several identified sub-attacks, which are aimed at sub-goals.

In the analysis performed here we will demonstrate this decomposition perspective of attack tree analysis. The possibility to design and analyze attack trees at different levels of depth and for selected areas of interest and criticality will be used in the analysis presented in this chapter and the following Chapter 6.

5.1 Selecting Critical Assets for Analysis

Based on the identification of the most important assets in the system, in the preceding section 4.8, we start the attack tree analysis by selecting some critical assets in the system to be the focus of analysis. As a criterion for selection of assets, we have chosen to look at the business processes and the services that the system application is offering, namely a distributed Internet-enabled calibration service. The question to answer has been: Which are the main assets in the system that would be critical for the possibility to offer a service, and what will be the main concerns for involved parties, i.e customers and Justervesenet, to actually want to use the services and to be confident regarding security concerns?

5.1.1 Metrology Related Assets

Some metrological related assets were identified in sections 4.8.1 - 4.8.3. From a metrological point of view, this service will have no value if the calibration results in the resulting calibration certificates are not correct, or if either customer or authority could mistrust the correctness of the values. For the customer, the calibration would not be valid, and when the calibrated instrument is used as a standard for calibration of other instruments, these calibrations would not be valid either. For the authority, it would not be possible to sign a calibration certificate if there are any doubts whether the values are correct. And for accreditation authorities it would be impossible to accredit a service where there is no clear certainty about the correctness of the calibration values. In other words, the correctness of calibration results is a very critical business process asset.

The other two most critical assets in the system from a metrological viewpoint would be the instruments in the calibration measurement procedure setup, customer's instrument(s)and authority's instrument(s). Both for customers and authority it will be vital to protect their instruments against unwanted influence and any usage that could cause damage to any of the inherent components. For the corresponding calibration results, it

will also be vital that the instruments are correctly set up and that connections are made according to measurement procedure specifications.

Critical for application, protecting the metrological related assets:

- Correct calibration values in calibration certificate (only valid with correct measurement procedure applied, correct connections between instruments)
- Correct operation of authority's calibration standard (should not be damaged), valid measurement results
- Correct operation of customers instrument (should not be damaged), valid measurement results

5.1.2 IT System Related Assets

Assets related to IT systems involved in the application were identified in sections 4.8.4 and 4.8.5. For Justervesenet, the authority in the system, it would be vital to not induce any unacceptable risk to the organizations IT system. Also the iMet application system itself should be protected, including software, hardware and information about the system.

For a customer to want to use the service, he would have to be certain that no unacceptable risk is implemented into his IT system. This certainty would be based on trust in the Justervesenet as the owner of the application system and provider of the service.

Critical for IT systems and owners of these:

- iMet application system, software and hardware
- Justervesenet's IT system, software and hardware
- Customer's IT system, software and hardware

5.2 Scope of Analysis

In the analysis performed here we have chosen to select different aspects of the iMet system and use these to investigate the feasibility and applicability of the attack tree method. This investigation has demonstrated various drawbacks and benefits with the methodology, as we will come back to in more details in chapter 9 where usability of the method is discussed closer.

In addition to selection of topics for analysis that could demonstrate different aspects of the analysis methodology, we have also selected topics based on the criteria that the topics' areas should be interesting, both in an information security perspective and in the perspective of metrology, specifically distributed Internet-enabled metrology.

The scope of this thesis work has therefore not been to perform a complete analysis of all possible attacks on the system. This would have been quite an extensive task, requiring resources beyond what is possible for this project in the form of time, personnel and available information. The result from such an analysis would be an extensive report with lots of details that would have limited interest for the purpose of this work.

As will be seen in the attack trees that we present here, the level of depth of analysis will vary quite a lot due to our scope of analyzing only selected areas. This in fact illustrates one of the advantages of the method, which is the possibility to specify sub-trees

at different levels of detail due to the necessity of prioritizing out of different concerns. In the attack trees this is observable when looking at the leaf nodes, which will vary a great deal in detailing level.

5.2.1 Focusing the Effort: Attack Goals for High-Level Analysis

For the analysis performed in this thesis work, we have chosen to focus on the criticality of correct calibration values in the calibration certificate. Analyzing the vulnerabilities in the system according to this has many interesting aspects in an information security setting as well. This application's functionality is based on a process where some code is downloaded to a customer. The code is signed at the database server, and this signature is verified at the customer to ensure the integrity of the code in the communication channel. The application is then run using this downloaded code, and measurement data is returned to an authority. The integrity of this measurement data, and the certainty about the origin of these data, that they are actually produced in a calibration procedure with the correct measurements involved, would be vital to have trust in the system. And thereby trust in the integrity and correctness of calibration values in the calibration certificate.

The attack tree that is showing the vulnerabilities and possible attack paths for this is presented in Figure 20, and the different nodes, with connections and different considerations regarding these are elaborated in section 5.4.

The other main focus selected is the criticality of integrity of software in the system. An attack tree elaborating this at functional level (high-level) is shown in Figure 21 and discussed in section 5.5. To illustrate how attack trees could be used with different level of abstraction, only the branch exploring how integrity of iMet application software could be compromised is shown.

5.3 Drawing the Trees

The attack trees, shown in this section and the following section 6, where analysis is taken a step further in detail, are presented in drawings in the figures. These figures need a little explanation regarding the drawing 'syntax'. The trees have been drawn from top to bottom, starting with the identified attack goals, which are equal to the assets to be protected. The other tree nodes are placed hierarchically under the top node, and connected with lines to the node describing the preceding event in a chain of events leading to an attack. In this way nodes linked in a chain up to the top-level attack goal, represents chains of events leading to a possible attack.

Where two or more nodes in the tree are linked together upward to a node on the higher level in the tree, two different logical relations are illustrated in the trees;

- Where two or more nodes have links leading to the same node one step up in the hierarchy, a logical **OR** applies between the elements, eg. in the attack tree in Figure 22, *Obsolete version used* **OR** *Manipulated version used* results in *Wrong version of program used*
- Where two or more nodes have links leading to the same node one step up in the hierarchy, and an arc is connecting the lines, together with an 'and' notification, a logical **AND** applies between the elements, e.g. in the attack tree in Figure 22, both *Manipulated program in DB* **AND** *Sign code with valid key* would have to be true to get *Valid manipulated version in DB*.

The leaf nodes, on the bottom of the trees, with no nodes hierarchically placed underneath, represents nodes that either could not be divided or explained further, or where we for the purpose of this analysis have chosen not to elaborate further.

In the text describing the different trees, when referring to the nodes the text is emphasized. The logical relations between them are written in bold font.

The presentation of the trees could also have been expressed in an outline language format rather than in a picture format. This could have been preferable, especially for large trees. For this work we have chosen the picture format to get a better visualization and clearer demonstration of the flexibility of the method.

In addition to drawing of the trees, they are explained and to some extent elaborated further, considering some implications and possible inter-relations in the text. To keep a link between the trees and the textual explanations, the nodes from the trees are emphasized in the text and the logical operators are accentuate in bold font.

5.3.1 Timing Dependencies

The attack tree method includes the presentation of logical dependencies of nodes in the tree's hierarchies. Expressing timing dependencies is more difficult. Events and states expressed in the nodes are either satisfied or not in the modelled attack paths. The only timing related criteria that could be expressed is that a node in the tree should be satisfied before the node higher up could be satisfied. For **AND**-linked nodes this means that all the nodes linked with an logical **AND** should be satisfied before the node higher up could be satisfied. Any expression of inter-sequencing of the nodes will not be able to express.

5.4 Attack Tree: Incorrect Calibration Values in Calibration Certificate

Figure 20 shows an attack tree describing how attacks could lead to *Incorrect calibration values in calibration certificate*.

As described earlier, the impact of incorrect calibration values in the calibration certificate would be serious. For both customer and authority it is vital to have trust in the correctness of the calibration values.

In the attack tree we identified two possible branches leading to *Incorrect calibration values*. The nodes expressing these situations are based on the data input to the calibration certificate and are expressed as:

- *Incorrect values from data collection* **OR**
- *Error in calculations*

The situation of getting incorrect values is closely related to the data input and how the data are handled in the calibration process and data transferral. Faulty data input to the calibration certificate could be derived from anywhere in the data transfer chain, from collection of measurement result at customer to the transferral of calibration values into the calibration certificate. Much of the critical functions here will be in data communication situations.

Error in calculations would to a great extent come to happen if there are any faulty program functions due to bugs, manipulation or other misuse. We will come back to a further elaboration of this in section 5.4.2.

In the analysis performed here we have not investigated situations of forgery of the

calibration certificate itself. This is defined to be outside the iMet application system scope, actually a part of a traditional calibration administrative business process.

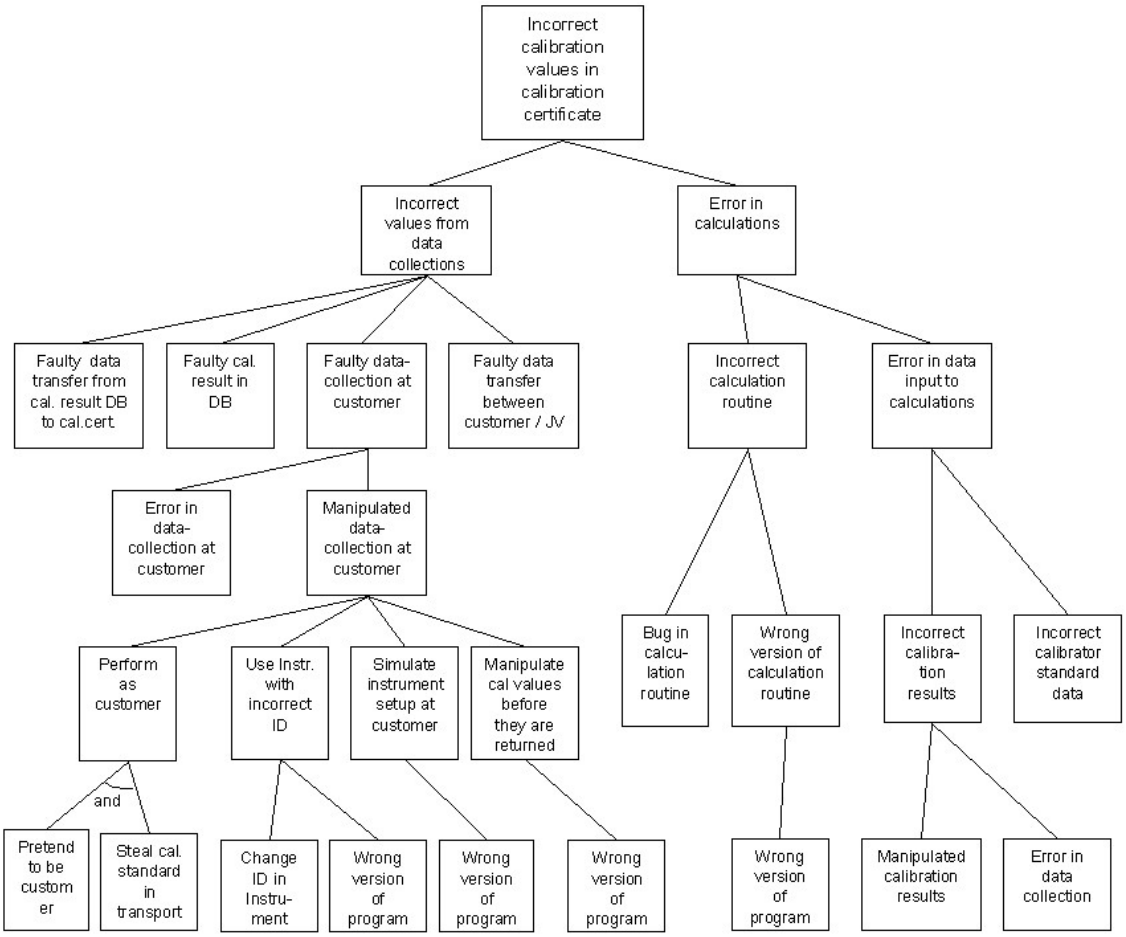

Figure 20: An attack tree showing possible attacks leading to incorrect calibration values in the calibration certificate

5.4.1 Incorrect values from data collection

There are four branches in the tree that possibly could lead to incorrect values. These are all all dealing with measurement data collection and transferral.

The nodes leading to incorrect values are:

- *Faulty data transfer from calibration result database to calibration certificate* **OR**
- *Faulty calibration result in database* **OR**
- *Faulty data transfer between customer and JV* **OR**
- *Faulty data collection at customer*

We have chosen to focus at one of the branches, that is *Faulty data collection at customer*. For the analysis of this specific system the vulnerabilities related to this are both interesting and critical. This part of the application functionality is vital for the system. In a traditional calibration procedure setup, the authority operator has the possibility to keep the measurements under surveillance, but with the distributed Internet-enabled application controlling the application, (s)he would now have to trust the application's functionality and security. This new situation brings possible new scenarios that should be investigated further.

In this analysis we have chosen not to elaborate the other three branches any further. It is nevertheless expected that further analysis could have revealed several possible vulnerabilities, but the nature of these would probably not have any specific interest from a metrological point of view.

The nodes leading to faulty data collection at customer are:

- *Error in data-collection at customer* **OR**
- *Manipulated data-collection at customer*

For the sake of this thesis work we have selected the latter condition for elaboration in further steps. Manipulated data collection at customer would have serious implications and as will be discussed later, motivation for manipulation could be present. Manipulated data collection at customer could come about in 4 different scenarios;

- Someone other than the actual customer could *Perform as customer* **OR**
- The actual customer could *Use instrument with incorrect ID* **OR**
- *Simulate instrument setup* **OR**
- (S)he could somehow *Manipulate cal values before they are returned*

For someone to be able to perform as a customer, two conditions must be met, the attacker must both:

- *Pretend to be customer* **AND**
- (s)he must actually have the calibration standard, this could be achieved by *Steal calibration standard in transport*

The calibration standard is sent to the customer together with a CD containing the main application for start-up and x.509 certificate with encryption keys for the customer as well as username and password for logging onto the server. The content of the CD is encrypted and decryption of this will require possession of the necessary password to decrypt this as well. Obtaining this (e.g. stealing or social engineering) will also imply that it will be easier for an attacker to impersonate as the customer.

We have identified two different ways for the customer to be able to use an instrument with incorrect ID in the calibration setup in a deceptive way:

- (S)he could either *Change the ID in his instrument* **OR**
- Use a *Wrong version of program*, which makes it possible to manipulate instrument ID

Measurement instruments today are not equipped with any secure digital identification tag. For instrument identification in the iMet application, only an instrument serial number placed in the instruments memory location is verified. This memory location is easily altered in most instruments on the market today. For some instruments there are actually no digital identification in memory, and the operator at the customer side should write the ID (serial number) down as a reply to the application.

To be able to simulate instrument setup at customer side a *Wrong version of program* would have to be implemented in the application system. By doing this the customer could actually run the application without having to connect the instrument to be calibrated in the setup. In this way (s)he could select preferred calibration result values for the instrument, and simultaneously have the instrument in normal production or use.

Another possibility for manipulation at the customer's side would be by *Manipulating cal values before they are returned*. Manipulation could be done by using a *Wrong version of program*. This would possibly imply a less extent of alteration than in the preceding scenario of simulation, but the implication of this would still be critical, as the values returned will not be correct for the calibration performed.

5.4.2 Error in Calculations

The data representing measurement results obtained from the calibration procedure would most often have to be processed in some way before the actual calibration values could be inserted into a calibration certificate. This processing could include formatting of data and calculations to correct for deviation or drift in calibration standards or measurement setup. Normally calculation of mean values and uncertainties related to each calibration value of an instrument will also be performed.

The vulnerabilities regarding calculation errors and data formatting errors are generally dealt with in traditional metrology system applications and we have chosen not to investigate details of these kinds of errors here. We will limit the description of these errors to a high level categorization shown in the attach tree. *Error in calculations* could be caused by either:

- *Incorrect calculation routine* **OR**
- *Error in data input to calculations*

An incorrect calculation routine could be caused by either:

- *Bug in calculation routine* **OR**
- Implementation of *Wrong version of calculation routine* into the system
  - This would be part of a *Wrong version of program*

Any error in the data input to calculations would be caused by either

- *Incorrect calibrator standard data* **OR**
- *Incorrect calibration results*, which could occur if either there are
  - *Manipulated calibration results* **OR**
  - *Error in data collection*

5.5 Attack Tree: Integrity of Software in the System

As discussed earlier, in analysis we wanted to focus on the criticality of protection of the assets related to IT systems, both for the application, Justervesenet and the customer. To support this we have chosen to investigate possible attacks leading to any compromising of integrity of software in the system.

Figure 21 shows an attack tree describing how attacks could lead to compromising the *Integrity of Software in the system*. Only the branch which explores how integrity of iMet application software could be compromised is elaborated, but the other two nodes, *Integrity of JV IT-system SW* and *Integrity of customer IT-system SW*, could have had similar tree structures showing possible attacks leading to these nodes. By elaborating the iMet application software branch we focus on how the application integrity could be compromised. Compromising the iMet application could have many implications, e.g. by inserting malicious software, this could lead to compromising Justervesenet's and customer 's IT systems as well.

5.5.1 Integrity of iMet Application Software

We have identified two different ways for an attacker to compromise integrity of the iMet application system. There could be:

- *SW deleted* **OR**
- Implemented any *Wrong version of SW* in the iMet application

The nature of this implemented software could be malicious and the insertion could be both intentional or unintentional, the difference of implications due to these two different situations is not explored any further here. It is though important to mention that in the attack tree method implementation we have chosen to regard both intentional and unintentional misuse as attacks to a system, and thereby the agents will be identified as attackers.

Deletion of software could also be either intentional or unintentional; also here the implications are not explored in detail. Software could be deleted at several locations:

- *In database* **OR**
- *In communication channel* **OR**
- *At customer*

The implications of scenarios where software is deleted would to a great extent depend

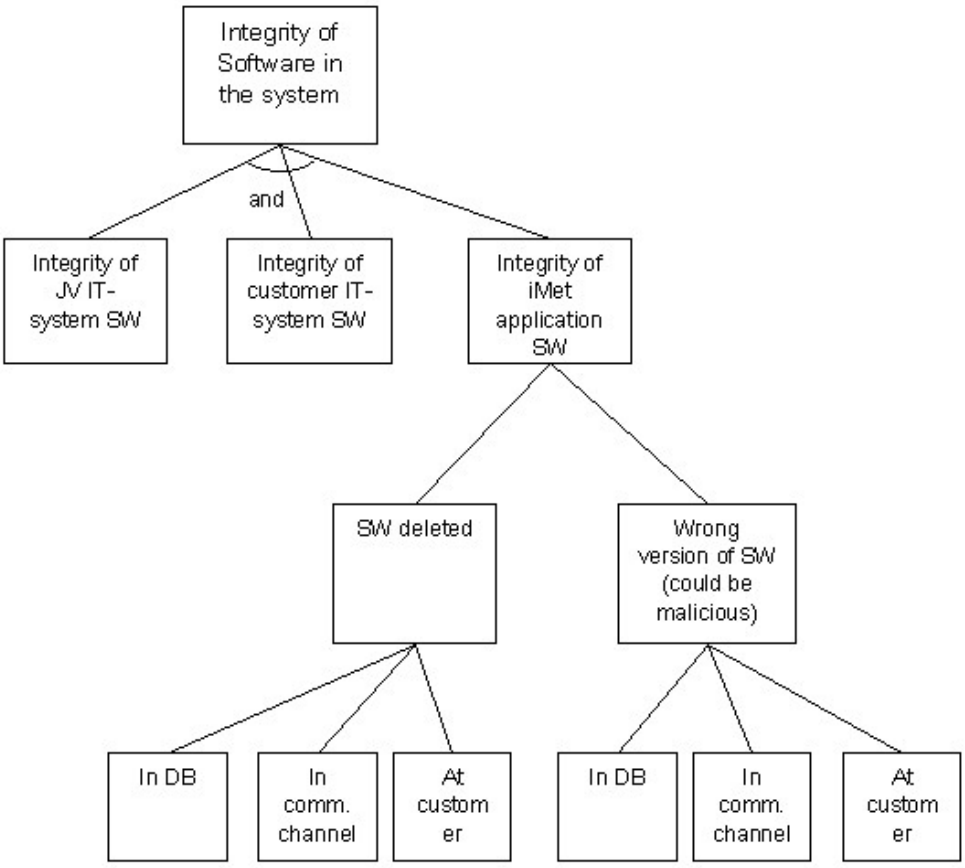

Figure 21: An attack tree showing possible attacks leading to compromising of the integrity of software in the system. Only the branch of integrity of iMet application software is expanded.

on backup routines and are not explored in any further detail here.

A wrong version of a software component could be implemented at several locations, This could be in:

- *In database* **OR**
- *In communication channel*
- **OR** *At customer*

In this high-level analysis we have investigated several aspects related to protection of critical assets in the system. The analysis is taken further in the next chapter. Here we will focus on a few areas of interest and we will for example look closer and explore in more detail how wrong versions of software actually could come be implemented and run in the system application. The elaboration of this is shown in Figure 22 in the next chapter.

6 Probing Into the iMet System

In this chapter we will explore further some of the possible vulnerable areas that were identified in the previous chapter 5. From the high-level analysis that were performed here we have chosen to go deeper into analysis of some aspects of the system.

We start by looking into how a wrong version of a program or program component could be implemented into the system. An attack tree illustrating how different ways to attack the system could lead to the fact that wrong versions of software are able to run, is shown in Figure 22.

In Figure 23, an attack tree is presented that goes even deeper into a selected section of this vulnerability area. Here we investigate possible ways for an attacker to access the database server. And in Figure 24 one section of the database server access tree is 'drilled' into by exploring how a hacker could get access to the database. The following sections will explain the trees in more detail.

In this stage of analysis we have followed the same criteria for selection of areas to analyze as in the earlier stages. One criterion in the selection of which topics to analyze has been the criticality for the system's business processes. The other criterion for selection has been how interesting the topic is for analysis. We have attempted to demonstrate analysis on several topics that have interesting implications for this specific system application for Internet-enabled calibration service. Nevertheless, the topics and areas analyzed here would have some general interest as well.

6.1 Analysis of the Attack Tree "Wrong Version of Program Used"

From the high-level analysis in the attack tree in Figure 20 at page 49, we found that several of the nodes were linked to equal leaf nodes; *Wrong version of program*. This indicates that if vulnerabilities could possibly lead to the use of wrong versions of programs or program components, this could have many different potential implications. How an attack, or chain of attacks, could result in the situation where the system is using wrong versions of any program component will be discussed further in the following analysis.

A correct version of a program will be the version that is implemented by an system administrator with designated responsibility and rightfully access to manage the system, and it will be the version that is developed, validated and maintained following quality assurance procedures and implemented maintenance and security policies. We have defined a wrong version of a program to be a version that should not be implemented in the system. As a result of this definition, program code with inherent (undetected) bugs will not be wrong versions. A correct version of a program will therefore not necessarily be a fault-free one, bugs could always be present in a correct version of a program in the way this is defined here.

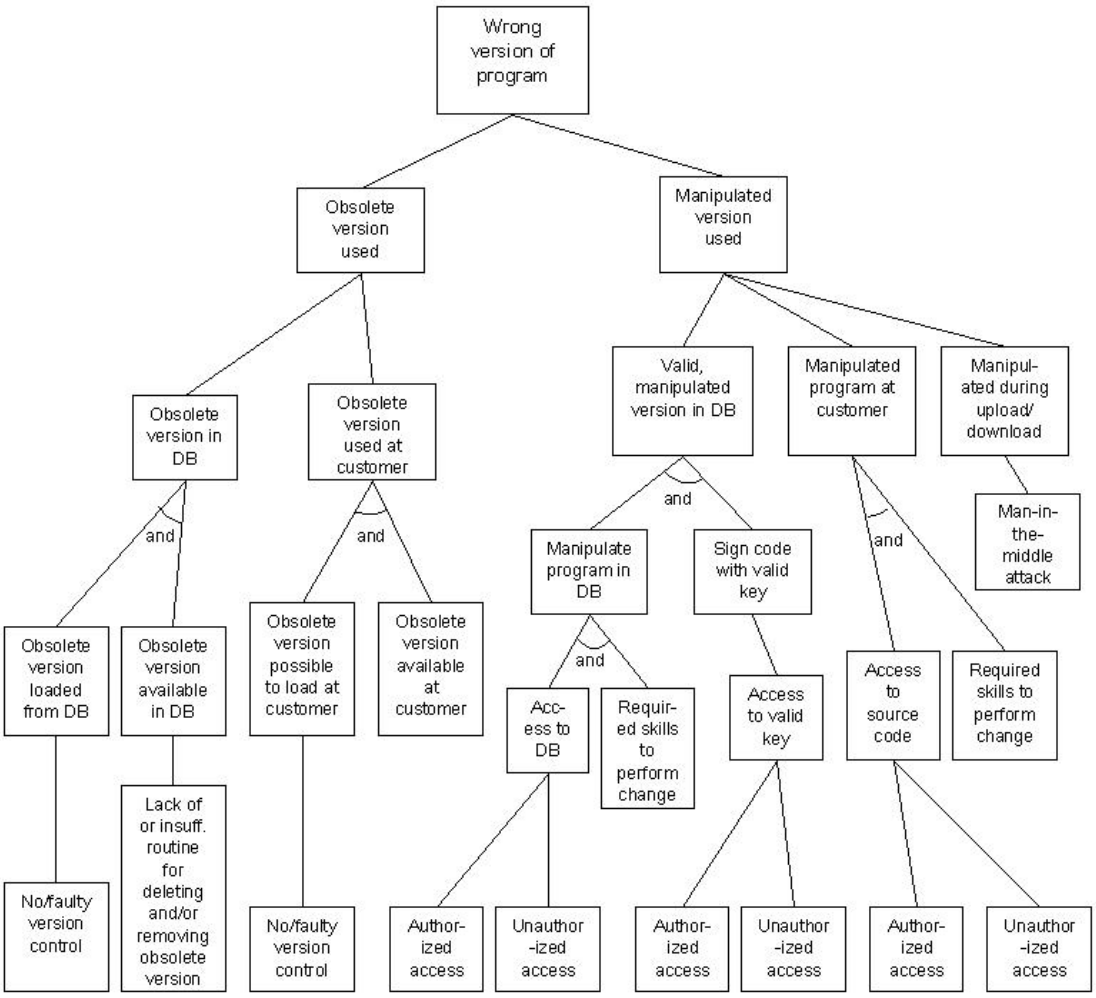

Figure 22: An attack tree showing possible attacks leading to implementation of wrong version of the system application program or program components into the system

When we look at the attack tree in Figure 22, we see that the top level node, *Wrong version of program*, has two different sub-nodes. A wrong version of a program could mean one of the two:

- *Obsolete version used* **OR**
- *Manipulated version used*

Both these two sub-node branches will be described in more details in the following sections 6.1.1 and 6.1.2. The possibility that a *Manipulated version* of program or component is used, could potentially have more serious impact than the possibility of an *Obsolete version* being used. This is discussed in more detail in chapter 8, where identified vulnerabilities are related to possible threats.

6.1.1 Obsolete version used

There are two branches in the attack tree that could lead to the use of an obsolete version in the system:

- Either an *Obsolete version in DB* is loaded and run **OR**
- An *Obsolete version used at customer*

As the following analysis shows, the latter situation does not necessarily mean that the program version used at the customer is one that is loaded from the database. We have though not assumed it possible that an obsolete version could be implemented into the communication channel, during upload or download of program components. Still, manipulated versions could possibly be implemented this way as we will come back to.

To run an obsolete version from the database, two conditions have to be met:

- There should be an *Obsolete version available in DB* **AND**
- It should actually happen that this *Obsolete version is loaded from DB*

Obsolete versions should of course not be stored in the database, but *Lack of or insufficient routine for deleting and/or removing obsolete versions* could lead to this. For an obsolete version to actually be loaded from the database, this will depend on the fact that there is *No or faulty version control* of the individual program components that are loaded and run.

The attack tree deals with the possibility that an obsolete version of the application, or individual program components, could be run at the customer. For this to happen, two different conditions would have to be satisfied:

- It should be *Possible to load obsolete version at customer* **AND**
- It should be *Obsolete version available at customer*

The possibility that an obsolete version is available at the customer is not analyzed further in the tree, but one could think of situations where the customer has used the service earlier, and that the older versions of the program components are not deleted from the customers computer.

The possibility that an obsolete version could be loaded (and run) at the customer will be present if there is *No or faulty version control*, which is the same condition as for the database loading possibility mentioned earlier.

Analysis of how an obsolete version could be used, shows that this is possible and could happen if the system lacks sufficient version control of the program components when these are loaded from DB, and when they are used at the customer site. Lack of or insufficient routines for deleting and removing obsolete versions of program components from the database could also lead to the possibility that obsolete versions are loaded and run.

6.1.2 Manipulated version used

If an attacker could initiate implementation and execution of manipulated versions of any program component in the system application, this could of course have several serious implications. We will discuss this further in chapter 7. The possibility for an attacker to initiate that a *Manipulated version is used*, is shown in three different branches in the tree. The three possibilities are:

- Either there could somehow be placed a *Valid, manipulated version in DB* **OR**
- There could somehow be implemented a *Manipulated program at customer* **OR**
- A program could be *Manipulated during upload or download*

To have a manipulated program component, that apparently is valid, inserted into the database, two conditions must be met:

- Someone should insert the *Manipulated program in DB* **AND**
- Someone must also *Sign code with valid key*

To be able to insert manipulated program components in the database, the attacker should have both:

- *Access to DB* **AND**
- *Required skills to perform change*[1]

Access to the database could be:

- *Authorized access* **OR**
- *Unauthorized access*

Abuse of authorized access to the database, in an attack aimed at implementing manipulated program components into the database, would have to be based on the attacker interacting with an insider or that the insider himself actually launches the attack. In section 6.2, possible attacks leading to unauthorized access to the database server are analyzed further.

To be able to sign the code with a valid key, the attacker should have *Access to a valid key*. This could be:

- *Authorized access* **OR**
- *Unauthorized access*

[1] 'Required skills' could in an attack tree analysis with attributes attached to nodes, be presented as an attribute. For the sake of simplicity in presentation we have chosen not to introduce attributes on nodes in the trees.

Authorized access to a valid key would in this situation imply that the attacker is an insider or interacts with an insider to launch the attack.

The possibility that program components could be manipulated (and later run) at customer, rest upon the facts that the person manipulating the code (the attacker) should have both:

- *Access to source code* **AND**
- *Required skills to perform change*

The application is built to check whether program components downloaded to the customer are signed using a valid key, but there is actually no application control of whether it is this specific code that is actually run, or whether the measurement results returned are generated by these program components. The implications of this are discussed further in chapter 7.

The access to source code could be either:

- *Authorized access* **OR**
- *Unauthorized access*

For the program components to be manipulated during upload/download, the attacker has to perform some kind of *Man-in-the-middle attack*. This possibility could have been elaborated further, but we have chosen not to go any deeper into this here. Man-in-the-middle attacks are discussed in several other works dealing with information security in distributed systems in general.

6.1.3 Leaf Nodes

In the attack tree, the elaborations of the different branches are worked out down to leaf nodes at different levels of the tree. As pointed out above there could for example be many attacker paths leading to a possible *Man-in-the-middle attack*, but we have chosen not to make any further investigations of this here.

For the other leaf nodes in the tree, we have done much the same considerations; The leaf nodes indicates the level of detail for the analysis performed here and the analysis could have been worked out further. To increase the detail level in analysis for most of the nodes, technical and possibly compromising details about the system should have been exposed, which could be undesirable.

Some of the leaf nodes could be argued to be atomic though, *Authorized access* to database, source code and valid key for signing of code is normally only possible to get following authorization procedures. Nevertheless if we analyze these situations in detail, there could be different ways to get authorized access as well. If there are any deceptive motives behind this, and the attacker finds a way to by-pass authorizing procedures, this access could normally be defined to be unauthorized.

There are three sets of apparently similar leaf nodes, *Authorized access / Unauthorized access* in the attack tree. These are linked to different nodes and even though they look identical in the tree, they do not denote the same situations. *Authorized ac-*

cess to database, source code or valid key for signing of code, would imply independent authorization for the three. *Unauthorized access* will likewise possibly require different approaches for an attacker.

In the following section we have probed deeper into the situation of one of the leaf nodes, *Unauthorized access* to the database server.

6.2 Analysis of the Attack Tree "Unauthorized Access to Database Server"

The implications of an attacker accessing the database server could be quite severe. The database server holds all application software. The private key used to sign source code, in order to secure integrity of source code in the communication channel to the authority and customer, is also found here.

Figure 23 shows how an intruder could access the database server and thereby be able to manipulate software in the system. The attacker would also need the valid private key to be able to sign code correctly, but this key could be found in the application on the server. If an attacker gets access to the source code in the database server, (s)he could also possibly generate his own key-pair and use this in the application for signing and verification at the customer's side. Verification at the customer side would presume that the customer somehow receives and accepts the public key generated in this key-pair.

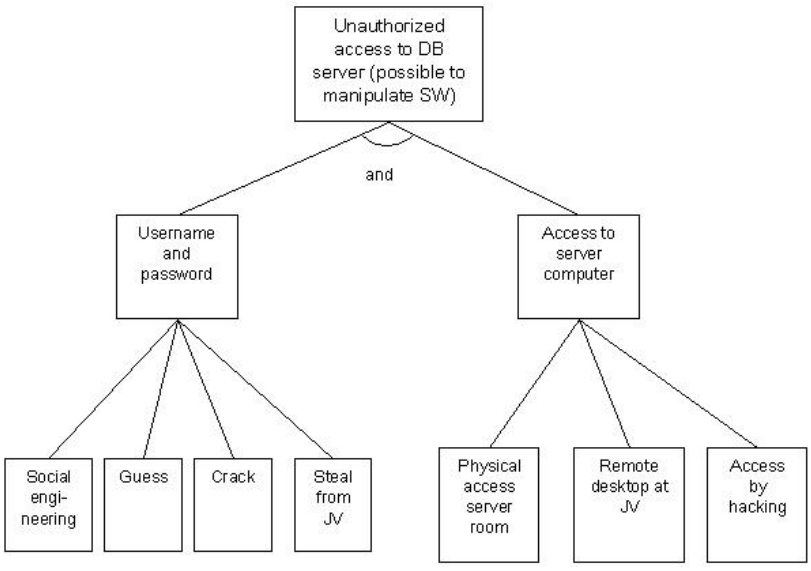

Figure 23: An attack tree showing possible ways to access the system database server, and thereby having the possibility to manipulate software in the system

As can be seen in the attack tree, *Unauthorized access to DB server*, access could be achieved if the attacker have both:

- *Username and password* for the server **AND**
- *Access to server computer*

6.2.1 Username and Password

The system application employs Usernames and passwords to login to the server when the application is started. To be able to obtain these an attacker could apply four different approaches, these would be:

- *Social engineering* **OR**
- *Guessing* **OR**
- *Cracking* **OR**
- *Stealing from JV*

Cracking and to some extent guessing could be possible if the passwords (and usernames) are weak. Social engineering and stealing implies that an attacker should get access to premises or get in contact with personnel at Justervesenet that has access to the information of interest.

6.2.2 Access to server computer

Access to the server computer could be obtained in three different ways, as shown in the tree. An attacker (which could of course also be an insider), could get:

- *Physical access to server room* **OR**
- (S)he could get access via *Remote desktop at JV*, as this is possible for computers in the JV IP-range **OR**
- An attacker with necessary skills and resources could *Access by hacking*

The other possible attacks, illustrated down to the leaf nodes, are also described in several other works. What is important for this system is to clearly show the risk that someone could actually access the database server, as the implications could be severe.

6.2.3 Access to DB server by hacking

All the bottom leaf nodes in this attack tree in Figure 23 could of course have been elaborated further. To demonstrate an example on how one could chose to elaborate by 'drilling down' in an attack tree where appropriate, the attack tree showing how the database server could be accessed by hacking, and thereby possibly be compromised, is presented in Figure 24.

An attacker could 'hack' access to the database server by either of the following two:

- *Exploiting misconfiguration* in any system or software components at the server **OR**
- *Exploiting weaknesses (bugs)* in these

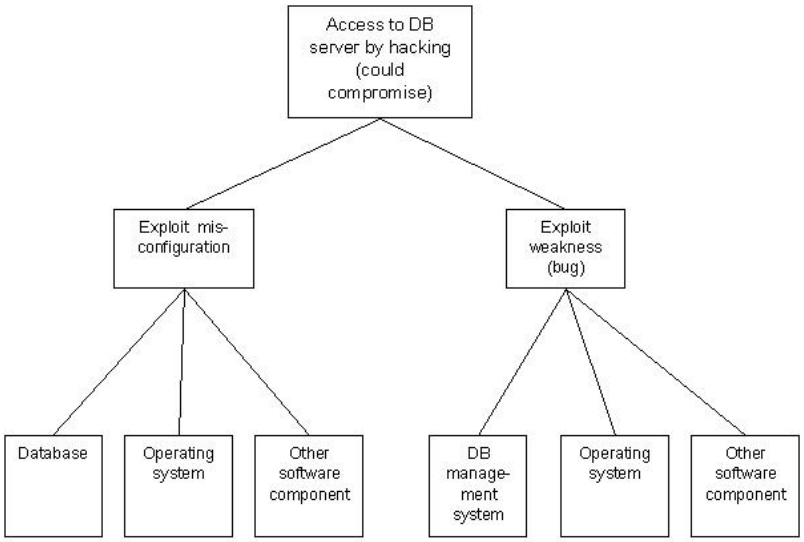

Figure 24: An attack tree showing possible ways to access the database server by hacking

Misconfiguration could exist in:

- *Database* **OR**
- *Operating system* **OR**
- *Other software component*

Weaknesses could have been implemented as bugs in:

- *Database management system* (DBMS) **OR**
- *Operating system* **OR**
- Any *Other software component*

To elaborate these vulnerabilities further in an analysis, we could have chosen to go a step even deeper and look at different weaknesses in e.g. the DBMS. To do this, different technical details and information should have been made available and explored.

6.3 Other Vulnerabilities

The attack tree method, as it is used in this analysis, gives the analyst(s) the possibility to select areas of interest and criticality and drill down into these areas in desired detail. This could of course imply omitting possible vulnerabilities, but this qualitative and creative approach also provides the possibility of focusing analysis effort into the most critical areas. The quality and accountability of the analysis results would of course have to rely on skills and experience of the analyst(s) and how these are used to make the best choices when prioritizing.

In the course of an analysis some vulnerabilities could be exposed when combining

results from the different attack trees and looking into relations and dependencies in the tree structures. For large systems and in deep-level analysis these relations and dependencies could of course be difficult to detect, and one would have to rely on more formalized and possibly automated methods to become aware of them.

In an analysis on the level we have performed here, some vulnerabilities could be detected in a more creative way. Based on the gathering of information and the analysis steps, different concerns becomes quite obvious and should of course be dealt with.

An example of this, in the iMet system, is the hardware configuration of different modules of the application. For the time being the database server is placed on the same computer as the communication server and in fact also together with the administrative application, which is used to maintain the system. This of course indicates serious implications regarding the vulnerabilities in the system. We have chosen not to elaborate these any further in the trees, as it is expected that the present setup will be changed once the system is put into productive use. But in the system setup implemented in the future, it will be reasonable to evaluate these vulnerabilities in coherence with the vulnerabilities identified here, as these will be similar.

The vulnerabilities identified in the analysis are presented and discussed in more detail in the following chapter 7.

7 Identified Vulnerabilities

In this chapter the vulnerabilities identified from analysis in chapters 5 and 6 will be discussed in more detail.

It should be pointed out that the attack trees and hence also the vulnerabilities identified must be considered merely examples. The focus of the present dissertation has been an investigation into the applicability of attack trees as an analytical tool and the results should not be considered an exhaustive risk and vulnerability analysis of the iMet system. However, the attack trees chosen in the analysis are dealing with critical business processes, and this means that most vulnerabilities described here will be critical to the system.

Identification of vulnerabilities, and the following descriptions of these, follows to a great extent what was read directly from the trees. Some were discovered in high level analysis and will be described her in this context (section 7.1. Some are described as results from deeper analysis (sections 7.2 and 7.3) and some were also identified as a result from the analysis process, even though not directly 'read' from the trees (section 7.4).

In chapter 8 some suggestions on how to deal with these vulnerabilities are presented. The identified vulnerabilities are evaluated together with possible threats to the system. The actual risk level will be a result of both possible vulnerabilities and possible threats to the system, and could not be elaborated directly from attack trees as they are presented in the analysis in chapters 5 and 6.

7.1 Vulnerabilities Identified in Attack Tree "Incorrect Calibration Values in Calibration Certificate"

In this section, vulnerabilities that are identified from the attack tree "Incorrect calibration values in calibration certificate", in Figure 20, are discussed.

7.1.1 Instrument ID could be Manipulated by Customer

Customer could manipulate instrument ID and use this to simulate measurements for another instrument than the one actually connected in the measurement setup. This 'method' could be used if the customer owns several instruments of the same type, and knows that one of them is 'good', or wants to avoid for some reason, convenience etc., to pick the instrument that is to be calibrated out of the setup where it is normally used. Instrument ID would normally be the serial number of the instrument, which is written in the instruments memory, and this could normally be easily altered.

Manipulation of instrument ID could also possibly be done by implementing manipulated software at the customer, to make it possible to alter the instrument ID to be returned to Justervesenet. The vulnerability of code manipulated at customer is also described in section 7.2.1.

7.1.2 Someone could Perform as a Customer

The attack tree shows that if someone could steal the calibration standard in transport and pretend to be the customer, this implies that the attacker uses a x.509 certificate

in the same sending and steals, receives or knows the username and password as well. The calibration results sent to JV would then be incorrect. This is an example of a vulnerability that shows a possible attack, but it is difficult to think of situations where an attacker should have any motivation to do this. One could 'invent' a situation with competing customers wanting to harm each other, by controlling the calibration values. This is an example of a vulnerability that has no obvious threat linked, and thereby no risk is actually imposed by this vulnerability.

7.2 Vulnerabilities Identified in Attack Tree "Wrong Version of Program Used"

In this section some of the vulnerabilities that are identified from the attack tree "Wrong version of program used", in Figure 22 are discussed.

7.2.1 Code Could be Manipulated by Customer

One identified vulnerability shows that it could be possible for a customer, with the required skills, to perform changes in the source code components that are downloaded to the customer's computer by the application. Source code components that are downloaded are signed in the database server and this signature is verified in the client part of the application before the code is compiled and run at the customer site. But it is not possible for the authority to be sure that the code that is compiled and run in the application actually is the same code that was downloaded and verified.

A skilled customer with access to the source code, which (s)he would have if (s)he had access to the computer, could start the application and wait till the code is downloaded. Then (s)he could for example imitate a hang-up in the application and then implement changes to the code, or implement code that already is changed. After initiating the calibration procedure once again, (s)he could then have the possibility to run manipulated code in the application.

The implication of this vulnerability would be that the customer might manipulate or simulate measurements, and thereby the measurement results. Simulating measurements could make it possible to run the calibration procedure without actually connecting any instrument to be calibrated in the setup. Manipulating the measurements could make it possible to directly manipulate undesired results or e.g. manipulate the instrument ID.

7.2.2 Code could be Manipulated in the Communication Channel

The attack tree also shows the vulnerability implicating that code could be manipulated via download/upload in the communication channel. This is not elaborated any further down than to be associated with a man-in-the-middle attack.

To be able to do this kind of attack, the attacker has to impersonate a customer when communicating with the database server, and impersonate a database server when communicating with the customer. To be able to actually insert code that is accepted by the application, the attacker would also have to sign the code in a way that will be accepted by the application in the verifying process. This could be done if the attacker could get access to the private key used to sign code. The IP-address of the server could be found in the configuration file that follows the application software. This is distributed to the customer on a CD together with the calibration standard, and is therefore not very difficult to find for an attacker that has a clear intention to do this.

The implications of this kind of code insertion could of course be quite severe. A hacker could insert malicious code like e.g. trojans, which could be run at customer.

A man-in-the-middle attack could also be directed to the communication channel set up between authority and database server. This would have much the same implications.

7.2.3 Code could be Manipulated in DB

How the database could be accessed is discussed section 7.3. If an attacker actually accesses the database server and the database, and has the necessary skills (and motivation), (s)he would be able to manipulate code and also delete code. The private key used to sign code is for the time being kept in the application management system, located at the same computer. The attacker could thereby use this key and sign any code that is implemented with a valid key. The implications of having valid manipulated code in the database could be quite severe. This could lead to the compromising of most of the SW assets in the system, obviously the iMet system, but also the customer and authority IT systems, as malicious code could be run at their computers. VISA instrument drivers that are run by the iMet system require administrator privilege to be set on the computer, so the malicious software would have every privileges at the customer's computer.

7.2.4 Obsolete Version Could be in DB

The attack tree shows that the presence of obsolete versions could be a vulnerability. The implications of this would primarily be that the integrity of the iMet system could be compromised, and that this obsolete version could be used at the customer, leading to possible faulty instrument operation or data handling.

In this analysis, version control routines and routines for deleting or removing obsolete versions are not evaluated and we will not investigate any further to what extent this vulnerability is actually present in the system.

7.2.5 Obsolete Version Could be Used at Customer

This vulnerability has much the same considerations as the previous. In addition to the lack of routines in the database management system, this vulnerability could also be present if the customer has run the application earlier and obsolete versions of the program components still are present on the customer's computer.

7.3 Vulnerabilities Identified in Attack Tree "Unauthorized Access to Database Server"

In this section, vulnerabilities that are identified from the attack tree "Unauthorized Access to Database Server", in Figure 23, are discussed.

7.3.1 Insider could manipulate code in DB

The attack tree shows that it could be possible for an insider (someone at JV, primarily someone with necessary skills and access) to insert, manipulate or delete code in the database. An insider should of course have necessary motivation to do this. The implications of this would be much the same as discussed in the preceding sections with the implementation of manipulated, possibly malicious code. In the case of an insider accessing the database, the actions performed could of course also be unintentional, but still cause much damage.

7.3.2 Hacker Could Manipulate Code in DB

The attack tree shows how it could be possible for a hacker to insert malicious code or delete code in the database. A hacker should also have necessary motivation to do this. The implications would also in this situation be much the same, but if a hacker gets access and thereby the ability to manipulate the database, this would most certainly be by inserting some kind of malicious code.

7.4 Other Identified Vulnerabilities

Here we mention several vulnerabilities, or vulnerable areas in the system, which are identified in the analysis as a result of the information explored and experience gained in the analysis. These vulnerabilities are not necessarily linked directly to the attack trees elaborated in the preceding chapters, but we have chosen to include them in the analysis nevertheless, as the implications could be severe.

As mentioned earlier we have had to make some assumptions regarding the implementation of the system. Still we have not made detailed assumptions regarding some of the topics that are considered as vulnerabilities in this section. For example there is no implemented clear policy for issues regarding certificate and key revoking, customers responsibility for components in the system like software and distributed certificates and keys. There has also not been implemented routines and assigned responsibility concerning further development and maintenance at Justervesenet. Some decisions should also be made regarding how the system is implemented in detail with connections to other parts of Justervesenet's IT-system. Issues to consider here are for example logging, backup, access and privileges.

The vulnerabilities mentioned here should therefore be considered, and possible implications in different scenarios elaborated further, prior to definite decisions regarding these issues are made and strategies implemented.

- For the time being the database server is installed at the same computer as the communication server and in fact also together with the administrative application, which is used for development and maintenance of the system. This fact will of course indicate serious implications regarding the vulnerabilities in the system. We have chosen not to elaborate these any further here, as it is expected that the present setup will be changed prior to putting the system into productive use.
  But the fact is important to mention, as it will be important to consider possible implications when the setup is being changed.
- In the present system management setup, certificates are issued by Justervesenet and all responsibility for the handling of certificates and keys are located here. No trusted third party (TTP) management is implemented in the system at the moment. Evaluation of possible implications regarding issuing, revoking and distribution lies with Justervesenet. And Justervesenet is also responsible for evaluation of strength in algorithms and key-sizes in the application. Customers would have to trust JV for the security in the application and the implementations of this. Key handling and the chaining of trust is not clear as the system is managed today. And some decisions are not definite about these issues. There are no definite decisions about how the keys should be distributed or any implemented requirements regarding responsibility, storage and revoking at customers. Elaboration of these issues should lead to

development and implementation of policy and effective routines taking care of this.

- The issue of trust will also affect the relationship between customer and Justervesenet in another fashion. The security level of the implemented application will to some extent have to depend upon the security level at customer's site. Implemented security policy and security mechanisms as up-to-date firewalls and virus protection in the customer's IT-systems as well as access restrictions will be vital, as well as procedures and restrictions regarding handling of security related credentials for the iMet system application.
- The server implementing most of the system components at Justervesenet is today possible to access via e.g. remote desktop at Justervesenet. To access it is only necessary to get hold of username and password. An insider at Justervesenet could of course quite easily get access to most of the system, with all possible implications. Even though the threat of someone at JV having the necessary motivation to compromise the system in a malicious way is considered to be small, the vulnerability should be considered.
- The system has no implemented policy, with routines and assigned responsibilities, for maintenance and further development of the system. Several implications regarding this should be elaborated further prior to putting the system into commercial use.

8 Recommendations, Mitigation Strategies

In this chapter we discuss some possible strategies to mitigate the vulnerabilities identified in chapter 7. To some extent we have also suggested prioritizing of the different protective measures and recommended actions. The evaluation of priority has been based on evaluation of criticality and some cost-benefit estimates.

The attack tree method has been used in this thesis as a means to identify vulnerabilities in the iMet system.

In the attack tree analysis, evaluation of criticality for business process and assets was a part of the initial selection of topics. We selected topics for analysis based on specific interest and or criticality, and from chapter 7, where vulnerabilities were identified, we therefore have got quite a good 'picture' of the most important vulnerabilities. Nevertheless, it has to be pointed out that several assumptions and limitations in the analysis, as described in section 4.6, means that this picture could not be an exact one, describing all possible vulnerabilities in the system. The recommendations for possible strategies for mitigation of identified vulnerabilities will therefore not be guaranteed to address every vulnerable area.

8.1 Threat and Risk Analysis

A risk analysis of a system will, as mentioned earlier and expressed in standards like [59] and [60], include several steps;

1. Identification of assets to be protected.
2. Identification of vulnerabilities
3. Identification of threats
4. Evaluation of risk, based on an overall judgment of vulnerabilities and threat.
5. Identification and proposal of counteractive measures to reduce risk

To be able to assess risk levels for the iMet system, some threat analysis should be performed. In this thesis work, a fully worked out risk assessment has not been a main objective, but we still want to demonstrate how the attack tree method could be used as a tool in the process to assess risk. In this way it will be possible to recommend strategies and measures to be implemented in the system to effectively enhance security.

8.1.1 Threats to the iMet System

The metrology business area is not a highly profiled business area, with large companies, big budgets and many people involved. The direct economical impact of errors will therefore not necessarily be very large. But for businesses depending on the services of metrology, and correct, reliable and traceable measurements, the consequences upon errors in measurements could be critical, both economic and otherwise.

Metrology is therefore an area where trust in reliable results and in the chaining of traceability is a key property. The main product of the business is correct results, including quantifiable uncertainty. Any error in a calibration of an instrument will propagate

in the traceability chain and the consequence could be critical for the end user of an instrument in the metrological hierarchy.

An illustrating example could be to consider the implications in e.g. the aviation industry. If the measuring instrument that is used to check some critical parameter in production of for example a navigation system is not making correct, reliable measurements, the consequences could of course be devastating.

8.1.2 Possible Attackers and Threats

To perform an evaluation of possible threats to the system one has to identify possible threat agents.

- **Customers** are expected to have low to medium motivation to attack the system. Today most customers are known personally to those responsible for calibration activities at Justervesenet. Also, customers would normally have interest in protecting their instruments and attaining reliable correct results for their instrument. But as the metrology area implements new technology, traceability will be disseminated in a new fashion. The customers could detect possibilities to misuse the systems to their own advantage.
- For **Insider**s, these are employees at Justervesenet, mainly the ones having access to some of the components in the system, motivation for attacks are expected to be low. The most valid threat could be unintentionally introducing vulnerabilities in the system; bugs, misconfiguration or disclosure of information. It should me mentioned that wages in the area are relatively low, if someone with economic resources has any interest in using an insider to perform changes, access system etc., it should be considered as a possibility that employees could be disposed to accept bribe. This will of course be the same for the systems that are used today. An insider could e.g. issue a fictitious calibration certificate.
- **Hacker**, i.e. outside attackers, are expected to have medium motivation to perform attacks. The Internet is known to be a highly insecure medium, with a constant threat of attackers wanting to induce malicious software, gain access to computing resources or use others servers as centres or intermediate point for other forwarded attacks etc. Attacks could be motivated and directed at the specific goal or more randomly effectuated. It is not expected though that Justervesenet or customers are more exposed to these threats than others, and this also applies to the iMet system application. For both Justervesenet and customers it is also to be expected that standard security strategies and measures are implemented, and these threat agents introduce therefore no additional threats to the system components.

8.1.3 Other Considerations, Future Use

The iMet system is developed as an application tool to support calibration services performed with Justervesenet or other NMI as the authority in the calibration setup. This means at high level in the calibration hierarchy.

An attacker with motivation to alter some specific measurement unit related to an end user measurement situation, would not be expected to do this high up in the calibration hierarchy. It would be more likely that effort and manipulation of measurements is performed close to this end user situation. Therefore the iMet system could not be expected to be prone to these kinds of attacker situations in present use of the system.

But even though the iMet system, as it is profiled today, is aimed for high-level calibration setups, the technology could be expected to be implemented in more low-level calibration setups in the future. Some general concerns for security, and protection against directed attacks, would therefore be valid.

8.2 Prioritizing Vulnerabilities to be Addressed

Table 2 at shows the evaluation of threat and risk level associated with the identified vulnerabilities in our analysis. Here is also presented possible mitigating actions or mechanisms to be implemented, based on the assessed risk levels and cost-benefit estimates.

We have used a plain three-level scale to express criticality of vulnerabilities, threats and risk level. Here is an explanation of the ratings for the three levels, covering the three different aspects that are evaluated in the table:

- **Low** - insignificant, not necessary to address
- **Medium** - considerable, needs attention
- **High** - serious, must be dealt with

The cost-benefit estimates that have been made here are not very accurate and were mostly based on the reasoning that if a countermeasure is easily implemented and has low cost attached, then it should be implemented even if the risk level associated is low or medium. If the risk level is high, countermeasures involving high cost and/or high effort should still be implemented.

8.3 Managing Risk

The risk analysis has provides us with an overview of some of the vulnerabilities in the system, together with estimates of the risk levels associated. Based on this list we have worked to come up with some recommendations for possible countermeasures or mitigation strategies. These are presented in section 8.4.

For some of the vulnerabilities it will be quite straightforward to express what would be the appropriate countermeasure for the threat related, this would especially be valid where administrative effort is required. It should be mentioned though that a suggestion for an administrative mitigation strategy, e.g. "Implement maintenance procedures", will have little value if the procedure implemented does not address the necessary critical functions in an appropriate manner.

Where more technical countermeasures are required, it would often require technical skills and thorough understanding of the system to be able to come up with a recommendation and implement appropriate countermeasures.

When implementing countermeasures and mitigation strategies into a system, it will be important to address the fact that this mitigation by nature could insert new vulnerabilities into the system. It will therefore be important to work toward an understanding of how this mitigation could actually change the system, and another cycle of risk analysis and mitigation should be performed.

It will also be important to address the dynamics of changing IT systems and surroundings together with the associated threats valid at the time of analysis. The fact is

Vulnerability	Criticality	Threat	Risk level	Mitigation
Instrument ID manipulated	High	Low	Medium	Camera surveillance Contract between customer and authority
Perform as a customer	Low	Low	Low	No suggested
Code manipulated by customer	High	Low/ medium	Medium	Code obfuscator Contract between customer and authority
Code manipulated in communication channel	Low	Medium	Low/ medium	Standard security mechanism implemented
Code manipulated in DB	High	Medium	High/ medium	Standard security mechanism implemented
Obsolete version in DB	Medium	Medium	Medium	Maintenance procedures
Obsolete version used at customer	Medium	Medium	Medium	Maintenance procedures Contract between customer and authority
Insider manipulates code in DB	High	Low	Medium	Maintenance procedures Responsibilities defined
Hacker manipulates code in DB	High	Low	Medium	Standard security mechanism implemented
Many components in system located at one computer	High	Medium	High/ medium	Split functionallity
Insider access possibilities	Medium	Low	Low/ medium	Maintenance procedures Responsibilities defined
Issues regarding trust	Medium	Low	Low/ medium	TTP implementation Contract between customer and authority
Key handling	Medium	Low	Low/ medium	TTP implementation Routines for key handling
Lack of routines for maintenance and development	Medium	Medium	Medium	Maintenance procedures

Table 2: Vulnerabilities, threat, risk level and possible mitigation for the iMet system

that risk analysis and effective mitigation will only be valid for the actual system analyzed in the characteristic surroundings when analysis were performed.

As recommended in standards like [59], a system should be protected by introducing risk management for systems and business processes, based on steps of risk analysis and implementation of countermeasures. Risk management should be a persistent process that goes beyond the risk analysis process and includes regular cycles of risk analysis and counteraction.

8.4 Some Recommendations

Here we have addressed the various identified vulnerabilities and we present possible strategies for mitigating of these.

8.4.1 Code Obfuscator

To prevent a customer from being able to manipulate or simulate measurements by manipulating program code, as described in section 7.2.1, a code obfuscator could be used. An obfuscator could be used as a tool to transform applications by renaming identifiers to meaningless characters, obfuscating metadata, and altering control flow so that the code is harder to understand. An example of a tool for this is Dotfuscator from Preemptive, [61].

Priority **HIGH**

8.4.2 Separate Functionality

To omit vulnerabilities due to location of several modules of the application on the same computer, described in 7.4, it would be a reasonable, easy implemented countermeasure to separate the functionality now maintained in the server computer to several computers, especially the administrative application, which is used for development and maintenance of the system, should be placed on another PC, protecting the accessibility. This separation of functionality is expected to be implemented prior to any commercial use of the system application.

Priority **HIGH**

8.4.3 Maintenance Procedures

To mitigate several vulnerabilities due to possible use of obsolete versions and availability issues, as described in sections 7.2.4 and 7.2.5, procedures for maintenance for the system should be developed and implemented. That is routines for back-up, implementation of version control, etc.

Priority **HIGH**

8.4.4 Routines for Key Handling

To mitigate vulnerabilities due to issues regarding key-handling, as described in section 7.4, policy and procedures for handling, issuing, storage and revoking of certificates and keypairs should be clearly defined and implemented for the trust chain between the parties involved. issue: lack of clear chaining of trust (i.e. ensuring that it's not just signatures verified here and there with gaps in between but rather that there's a chain of custody from JV to the customer and actual execution as well as return of the data)

Priority **HIGH**

8.4.5 Development and Implementation of Trust Chain

In the present system management setup, certificates are issued by Justervesenet and all responsibility for the handling of certificates and keys are located here. No trusted third party(TTP) management is implemented in system at the moment. Evaluation of possible implications regarding issuing, revoking and distribution lies with Justervesenet. And Justervesenet is also responsible for evaluation of strength in algorithms and key-sizes in the application. Customers would have to trust JV for the security in the application and the implementations of this. Key handling and the chaining of trust is not clear as the system is managed today. And some decisions are not definite about these issues. There are no definite decisions about how the keys should be distributed or any implemented requirements regarding responsibility, storage and revoking at customers. Possible vulnerabilities as consequences of these matters are also described in section 7.4. Elaboration of these issues should lead to development and implementation of policy and effective routines taking care of this.

The implementation of trusted third party (TTP) management could be one of the measures to attain trust chain in the system.

Priority **MEDIUM**

8.4.6 Camera Surveillance

For the authority to be able to have better control of the measurement setup at customer, mitigating vulnerabilities addressed in sections 7.1.1, 7.1.2 and 7.2.1 a web camera surveillance could be implemented in the system.

Priority **MEDIUM**

8.4.7 Responsibility Definition

To clearly state the responsibilities for the parties in a distributed Internet-enabled calibration process, a contract between customer and authority should be formulated and used. The contract should express different concerns regarding responsibility and possible liability for damages. The vulnerabilities that could be mitigated by this are described in sections 7.1.1, 7.2.1, 7.2.5 and 7.4.

Priority **HIGH**

8.4.8 Other Considerations

General concerns regarding network and Internet security clearly applies for a security analysis for the system. Vulnerabilities that are related to this are described in sections 7.2.2, 7.2.3, and 7.3.2. But these issues are not considered application specific and for the analysis in this thesis work they are not dealt with specifically. As general countermeasures, it is assumed that standard security mechanisms are implemented in the IT systems of both customer and Justervesenet.

8.4.9 Not Prioritized

In the table, where no suggested countermeasures are denoted, the risk is not prioritized.

9 Usability of the Method

Usability of the attack tree method has been demonstrated by this work and the results from it in terms of identified vulnerabilities and suggested counteractive measures to these vulnerabilities, signifying the methods applicability. In this chapter we will summarize our findings related to usability issues based on experiences from the analysis process.

9.1 Attack tree Method, Scope

When discussing the usability of the attack tree method, it would be appropriate to describe the scope of this method in relation to the processes associated with a general risk management methodology. As discussed in section 2.1, several standards and methods address risk analysis and management.

One standard that is often referred to, the ISO/IEC 27001 standard, [62], proposes a requirement specification for information security management systems (ISMS), defining essential components in such a system;

- **Security Policy**
- **Statement of applicability**, that is definition of what part of an organization or system the system applies to
- **Risk assessment**, that is identification of vulnerabilities and threats, and evaluation of their impact for the organization or system defined
- **Security Management**, that is the selection of actions based upon selected security level (implementation of security policy)
- **Requirements for Evidence**, that is documentation for selected actions and basis for audits

To be able to employ attack trees successfully in an analysis, a security policy for the organization and system should be implemented and understood. Documents and information defining the extension of system or specific application being analyzed, as well as necessary information about the system and related system, should also be made available. In this thesis we have had to state some limitations and make some assumptions as regard of these elements.

The basis for analysis would be an identification of the assets in the system or organization to be protected. This relates to the statement of applicability in the list of defined essential components in an ISMS.

The attack tree method clearly applies as a tool for identifying vulnerabilities in an organization or system. This is as described earlier as part of the risk assessment process, which also includes threat analysis. The actual risk assessment and risk management would be based on objectives defined in the security policy, and the selection of appropriate risk levels. Selecting appropriate countermeasures would be part of the security management process.

As we have seen in this work, the selection of mitigation strategies for the system is clearly related to the identified vulnerabilities, and the attack trees from analysis presents a picture that could be used to also identify possible mitigation strategies.

The attack trees developed in the analysis could, apart from being the basis for vulnerability analysis, also be used as documentation for the analysis performed and the selected counteractive actions. This documentation could as well be used as a foundation for further analysis in later stages of development for a system, or when changes are implemented in i.e. surroundings or assumptions regarding security level.

9.2 Formalized Method, Experiences

The attack tree method is a formalized method. The analysis process includes several steps, guiding the analyst to consider all assets and possible vulnerabilities and threats to these. This analysis could differ in depth and detail though.

The method is however understood as semi-formal, there is no strict formalism in the method as it is implemented here. This gives a flexible scope in the implementation of the method. It follows from this that there is no obligation to the level of detail for the analysis. For this type of system it would be possible to choose to initially perform high-level analysis and then dig deeper into selected areas. This scope for an analysis leads to flexibility in several directions.

The target for analysis in this thesis has been a system that actually has not been finalized. Several assumptions have had to be made in course of the analysis, partly because of lack of a clearly defined implementation of an application, but also because of limitations in available information and resources for the analysis itself. For this type of system and level of abstraction, the attack tree method has though shown to be very suitable. Even with the limitations mentioned it has shown to be possible to identify relevant vulnerabilities, leading to a good picture of security risks in the system.

For systems of larger size or complexity or when going deeper in detail and performing full analysis for a system, it should be mentioned that one probably should use some kind of automated tool to build the trees, and thereby probably also detect some dependencies not clearly visible for the analyst(s). Also, such an analysis would require collection of more detailed information, input from experienced analysts and input from databases for vulnerabilities etc.

In the course of an analysis some vulnerabilities could be exposed when combining results from the different attack trees and looking into relations and dependencies in the tree structures more creatively. For large systems and in deep analysis these relations and dependencies could of course be difficult to detect, and one would have to rely on more formalized and possibly automated methods to become aware of them.

Flexibility in the method, summary:

The flexibility in applying the method has in this work demonstrated different possibilities;

- *Flexibility for depth of analysis*, one could select different level of detail and choose to penetrate into areas of interest or criticality.
- *Flexibility for maturity of application*, one could use the method in different phases of development and implementation of a system.

- *Flexibility in interpretation of results and identification of vulnerabilities*. Some vulnerabilities could be detected by interpreting the results in a creative way. Based on the gathering of information and the analysis steps, different concerns becomes quite obvious and should of course be dealt with.

9.3 Presentation of Results

In the analysis process, some reflections about the presentation of the result in an analysis like the one performed here have become apparent, and should be presented.

The trees in the analysis give a good visible picture of how vulnerabilities apply, this is easy to understand and also for for non-professionals some of the inherent dependencies are easily understood. The trees could therefore be a good tool for presentation of the security analysis for a system.

When presenting the results one should nevertheless consider whom the presentation is for, as this pictures presented could be differently interpreted and understood. As for the analysis presented here the results could be communicated to two different groups;

- To executives, the analysis results could be presented as a basis for obtaining (financial or time) resources to implement necessary security mechanism. As the attacker point of view is presented, the trees automatically draw attention to security issues and could be an illustrative tool to support analysis presentation.
- To a customer different considerations should be made. The method focuses on showing possible attacks on the system. Attack trees shows possible attacks, and visually enlarges security issues by drawing attention to these attacks. The trees could thereby give an overrated impression of insecurity if other parameters in the analysis are not understood in the same fashion.
  Therefore the attack trees themselves would probably not be an appropriate attachment when presenting the system, and security in the system.

9.4 Implementation of Attributes in the Method

In the analysis process, we made some attempts to introduce attributes in the trees, particularly for attacker cost estimates and necessary skills for an attacker. It was very difficult to make sound estimates and effectively implement the attributes. Therefore, what we ended up with in a couple of the trees was the inclusion of separate nodes for necessary skills.

Especially, it was very difficult to make accurate and useful estimates of expected cost, and though it was difficult to get trustworthy results. Including attributes in an effective way could be applicable on small areas or possibly on selected topics. It is quite clear that the analyst(s) would need a lot of experience and/or data to do good estimates to support the attributes.

Evaluation and assessment of attributes would be quantitative analysis. In this work we have chosen to not look further into this, but several implications of this topic could be elaborated in further work.

10 Conclusion and Further work

The goal of this master thesis has been to perform investigation of the attack tree method, and to study some distinctive characteristics of this method. A main objective has been the demonstration of how the attack tree method could be applied as a means for analyzing the security level in a specific system.

We have implemented the method in a security analysis for a specific system application, the iMet system, addressing different security issues related to confidentiality, integrity and availability of the system. Following this attack tree analysis and derived identification of vulnerabilities, we have investigated possible threats to the system and performed an assessment of risk for the vulnerabilities. This analysis process has made it possible to identify and recommend possible countermeasures for the system.

The specific results from analysis, together with an evaluation of the usability of the method are the main contributions from our work.

10.1 Implementing the Method

The implementation of analysis was done in a process of several steps;

1. **Identifying critical assets**
   Initially some critical assets in the system were identified. The assets for consideration were selected based on prioritizing criticality for business processes and choosing topics of special interest.

2. **Analysis using the attack tree method**
   Analysis was performed using the attack tree method. The analysis was effectuated with differentiated level of detail. First some high-level analysis for assets of special interest was performed, then interesting and specifically critical areas were investigated further with a more detailed analysis.

3. **Identifying vulnerabilities**
   Based on the attack tree analysis some vulnerabilities in the system were identified. The vulnerabilities could be derived directly from the trees or identified as a result of evaluation of the information explored and experience gained in the analysis.

4. **Threat and risk analysis**
   Important threats to the system were discussed, in a scope of possible threat agents, and risk level for the identified vulnerabilities were estimated based on evaluation of these threats.

5. **Recommending mitigation strategies**
   From the identified vulnerabilities and accordingly assessed risk levels, some appropriate mitigation strategies for the system were recommended.

10.2 Major Achievements

The results derived from analysis have had two major achievements. First, the analysis has made it possible to accomplish definite results from the analysis of security for the iMet system, both the presentation of a picture of the security level in the system and the proposing of some appropriate strategies to enhance this security level.

One other result derived has been the evaluation of usability of the attack tree method for analysis of the specific system, and thereby also for other systems systems similar in size and complexity.

10.2.1 Results from System Analysis

In the analysis we identified critical assets and performed an analysis based on the attack tree method. This resulted in identification of several vulnerabilities to the system, the most important are presented here:

Identified Vulnerabilities

1. **Instrument ID could be manipulated** The customer could manipulate instrument ID and use this to simulate measurements for another instrument than the one actually connected in the measurement setup.
2. **Code could be manipulated by customer** It could be possible for a customer, with the required skills, to perform changes in the source code components that are downloaded to the customer's computer by the application.
3. **Code could be manipulated in communication channel** It was shown that code could be manipulated via download/upload in the communication channel. This could be between customer and database server or between authority and database server.
4. **Code could be manipulated in DB** If an attacker actually accesses the database server and the database, and has necessary skills (and motivation), (s)he would be able to manipulate code and also delete code.
5. **Obsolete version could be in DB** The attack tree analysis showed that obsolete versions of program code could be present in the database.
6. **Obsolete version could be used at customer** Obsolete versions of program code could be present at customer site, and possibly run in the application
7. **Insider could manipulate code in DB** The attack tree analysis showed that it could be possible for an insider (someone at JV, primarily someone with necessary skills and access) to insert, manipulate or delete code in the database.
8. **Hacker could manipulate code in DB** The attack tree showed how it could be possible for a hacker, with necessary skills and motivation, to insert malicious code or delete code in the database.
9. **Many components in the system are located at one computer** The database server is installed at the same computer as the communication server and the administrative application, which is used for development and maintenance of the system. This fact indicate serious implications in the system.
10. **Insider could access system components** An insider at Justervesenet could quite

easy get access to most of the system, with all possible implication to this.

11. **Issues regarding trust** In the present system management setup, certificates are issued by Justervesenet and all responsibility for the handling of certificates and keys are located here. Customers would have to trust JV for the security in the application and the implementations of this.
12. **Key handling issues** Key handling and routines regarding issuing, distribution and revoking of certificates and key-pairs are not clearly defined and implemented in the system today.
13. **Lack of routines for maintenance and development** Routines for maintenance and development and definition of responsibilities for the system are not clearly defined and implemented in the system today

Proposed Countermeasures

Following the attack tree analysis and identification of vulnerabilities, we investigated possible threats to the system and performed assessments of risk for the vulnerabilities. This made it possible to recommend some possible countermeasures:

1. **Implementation of code obfuscation** An obfuscator could be used as a tool to transform applications by renaming identifiers to meaningless characters, obfuscating metadata, and altering control flow so that the code is harder to understand, and thereby effectively manipulate.
2. **Separation of functionality onto different computers** To omit vulnerabilities due to location of several modules of the application on the same computer, it would be a reasonable, easy implemented countermeasure to separate the functionality now maintained in the of server computer to several computers
3. **Development and implementation of maintenance procedures** To mitigate several vulnerabilities due to possible use of obsolete versions and availability issues, procedures for maintenance for the system should be developed and implemented.
4. **Development and implementation of routines for key handling** To mitigate vulnerabilities due to issues regarding key-handling, policy and procedures for handling, issuing, storage and revoking of certificates and key-pairs should be clearly defined and implemented for the trust chain between the parties involved.
5. **Development and implementation of trust chain** Policy and effective routines for a trust chain between the parties involved should be developed and implementation. The implementation of trusted third party (TTP) management could be one of the measures to attain a trust chain in the system.
6. **Implementation of camera surveillance** For the authority to be able to have better control of the measurement setup at customer a web camera surveillance could be implemented in the system.
7. **Definition of responsibilities between authority and customer** To clearly state the responsibilities for the parties in a distributed Internet-enabled calibration process, a contract between customer and authority should be formulated and used. The contract should express different concerns regarding responsibility and possible liability for damages.

10.2.2 Usability of the Attack Tree Method

The evaluation of usability of the method was based on experiences from the analysis phase and the specific results derived. This had to some extent to be subjective experiences and thereby an evaluation out of subjective preferences. We have divided the usability criteria into three main groups:

Flexibility Scope

The attack tree method showed to be suitable for the purpose defined in this thesis work. The flexibility in the method made it possible to perform an analysis with valuable results even though there were obvious limitations in available resources and information for analysis. The flexibility in the method could be expressed in three different associations:

- **Flexibility for depth of analysis** The possibility to select different levels of detail for the analysis and chose to penetrate into areas of interest or criticality, makes the analysis method an effective means for analysis of systems where one could define specific critical areas.
- **Flexibility for maturity of application** The method could be applied for systems in different phases of development and implementation of a system.
- **Flexibility in interpretation of results and identification of vulnerabilities** Some vulnerabilities could be detected by interpreting the results creatively. Based on the gathering of information and the analysis steps, some resemblances might become quite obvious by observing the total picture. This understanding makes it possible to identify vulnerabilities in a more high-level scope.

Presentation of Results

The trees in the analysis give a good visible picture of how the vulnerabilities applies, this is easy to understand and also for non-professionals some of the inherent dependencies in the trees are easily understood. The trees could therefore be a good tool for presentation of the security analysis for a system. But one should adapt the presentation according to the recipients.

Implementation of Attributes

Some attempts were made to introduce attributes in the trees. It showed to be difficult to make accurate and useful estimates, and thereby effectively attach attributes to the nodes. Including attributes in an effective way could be applicable on small areas or possibly on selected topics where necessary data for the making good estimates is available.

10.3 Further Work

We have performed a study where one of the main objectives has been the evaluation of usability for the attack tree method in a risk analysis process. The evaluation has to a great extent been based on subjective experiences from the analysis process. To address this usability evaluation in a different manner it could be useful to initialize work that could verify or extend the scope of identified results. A comparative case study could possibly expand the perspective for experiences and possibly give supporting or diverting results. One could think of studies where the attack tree method were compared to either

a strictly formalized method, e.g. some kind of flaw hypothesis testing, a check-list based method or both.

Regarding the attack tree analysis method, one possible direction for further work could also be to look more closely into the employment of attributes for the type of analysis presented here, i.e. when combining high-level and low-level analysis, and whether the use of attributes could support the method by bringing extra information into the attack tree abstractions.

For Justervesenet, and other similar cooperative organizations within the metrology area, the experiences drawn from this thesis work could as well be applied for analysis of new systems developed, in different phases of development, or for implementation of extensions to other existing systems.

For the iMet system application, and for Justervesenet as the owner of this application, it would be appropriate to follow up this study with a more thorough analysis and a study of details in some of the areas that we have not looked closely into here. Security and risk management strategies could also be developed and incorporated more tightly into the processes and systems related, by introducing a more formalized risk and vulnerability analysis on the design and development mechanisms for this system.

Bibliography

[1] ISO/IEC. 1998. ISO/IEC TR 13335-3 Information technology - Guidelines for the management of IT Security - Part 3: Techniques for the management of IT Security.

[2] Leveson, N. G. 1995. *Safeware: System Safety and Computers*. Addison-Wesley, Reading MA.

[3] Viega, J. & McGraw, G. 2002. *Building Secure Software: How to Avoid Security Problems the Right Way*. Addison-Wesley.

[4] Richard R. Linde. 1975. Operating System Penetration. *Proceedings of the National Computer Conference*, 44. AFIPS Press, Montvale, NJ,.

[5] Mollatt, S., Kraft, J., Stranden, R., Svartvadet, J. O., Gulbrandsen, R., & Johnsen, O. Metoder og verktoy for gjennomforing av risikoanalyser. Technical report, ASIS Norway, 2002.

[6] Schneier, B. dec 1999. Attack Trees. Internet, http://www.schneier.com/paper-attacktrees-ddj-ft.html. Dr. Dobb's Journal (Visited May 2007).

[7] SAINT. http://www.saintcorporation.com/products. (Visited June 2007).

[8] BIPM. International Vocabulary of Basic and General Terms in Metrology, 1993.

[9] Justervesenet. http://www.justervesenet.no. (Visited May 2007).

[10] SofTools - Metronet. http://www.amctm.org/index.asp. (Visiyed May 2007).

[11] EUROMET. April 2005. Implementing Metrology in the European Research Area, IMERA. http://www.euromet.org/projects/imera. (Visited May 2007).

[12] Nov. 2005. NS-EN ISO/IEC 17025 General requirements for the competence of testing and calibration laboratories.

[13] Sand, A. *Applying the Internet to Instrumentation and Metrology*. PhD thesis, Norwegian University of Science and Technology, Trondheim, January 2007.

[14] Moore, A. P., Ellison, R. J., & Linger, R. C. Attack Modeling for Information Security and Survivability. Technical report, Carnegie Mellon University, March 2001.

[15] Mauw, S. & Oostdijk, M. 2006. Foundations of Attack Trees. *LNCS*, 3935, 186–198.

[16] Cynthia Phillips & Laura Painton Swiler. 1998. A graph-based system for network-vulnerability analysis. In *NSPW '98: Proceedings of the 1998 workshop on New security paradigms*, 71–79, New York, NY, USA. ACM Press.

[17] Steven J. Templeton & Karl Levitt. 2000. A requires/provides model for computer attacks. In *NSPW '00: Proceedings of the 2000 workshop on New security paradigms*, 31–38, New York, NY, USA. ACM Press.

[18] Dacier, M. *Towards Quantitative Evaluation of ComputerSecurity*. PhD thesis, Institut National Polytechnique de Toulouse, December 1994.

[19] Oleg Sheyner, Joshua Haines, Somesh Jha, Richard Lippmann, & Jeannette M. Wing. may 2002. Automated Generation and Analysis of Attack Graphs. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, 273, Washington, DC, USA. IEEE Computer Society.

[20] Paul Amman, Duminda Wijesekera, & Saket Kaushik. nov 2002. Scalable , Graph-Based Network Vulnerability Analysis. In *Proceedings of the 9th ACM conference on Computer and Communication Security*.

[21] Swiler, L., Phillips, C., Ellis, D., & Chakerian, S. June 2000. Computer attack graph generation tool. In *Proceedings of the DARPA Information Survivability Conference and Exposition*.

[22] Fred Cohen, Cynthia Phillips, Laura Painton Swiler, Timothy Gaylor, Patricia Leary, Fran Rupley, Richard Isler, , & Eli Dart. sept 1998. A Preliminary Classification Scheme for Information System Threats, Attacks, and Defenses; A Cause and Effect Model; and Some Analysis Based on That Model.

[23] Ronald W. Ritchey & Paul Ammann. 2000. Using Model Checking to Analyze Network Vulnerabilities. In *SP '00: Proceedings of the 2000 IEEE Symposium on Security and Privacy*, 156, Washington, DC, USA. IEEE Computer Society.

[24] Oleg Mikhail Sheyner. *Scenario graphs and attack graphs*. PhD thesis, Carnegie Mellon University, April 2004. Chair-Jeannette Wing.

[25] Jha, S., Sheyner, O., & Wing, J. 2002. Two Formal Analys s of Attack Graphs. In *CSFW '02: Proceedings of the 15th IEEE Computer Security Foundations Workshop (CSFW'02)*, 49, Washington, DC, USA. IEEE Computer Society.

[26] Ammann, P., Pamula, J., Ritchey, R., & Street, J. des 2005. A host-based approach to network attack chaining analysis. In *21st Annual Computer Security Applications Conference*, 10 pp. ISE Dept., George Mason Univ., Fairfax, VA, USA.

[27] Mehta, V., Bartzis, C., Zhu, H., Clarke, E., & Wing, J. Sept 2006. Ranking Attack Graphs. In *Proceedings of Recent Advances in Intrusion Detection 2006*, Hamburg, Germany.

[28] Daley, K., Larson, R., & Dawkins, J. aug 2002. A structural framework for modeling multi-stage network attacks. In *International Conference on Parallel Processing Workshops, Proceedings.*, 5–10, OK, USA. Tulsa Univ.

[29] Tao Zhang, Ming-Zeng Hu, Dong Li, & Liang Sun. Aug 2005. An effective method to generate attack graph. In *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, volume 7, 3926– 3931. Res. Center of Comput. Network & Inf. Security Technol., Harbin Inst. of Technol., China.

[30] Steven Noel & Sushil Jajodia. 2004. Managing attack graph complexity through visual hierarchical aggregation. In *VizSEC/DMSEC '04: Proceedings of the 2004*

ACM workshop on Visualization and data mining for computer security, 109–118, New York, NY, USA. ACM Press.

[31] Wing, J. March 2005. Scenario Graphs Applied to Security. In *Proceedings of Workshop on Verification of Infinite State Systems with Applications to Security*, Timisoara, Romania.

[32] Dantu, R., Loper, K., & Kolan, P. 2004. Risk management using behavior based attack graphs. In *Proceedings. ITCC 2004. International Conference on Information Technology: Coding and Computing*, volume 1, 445 – 449.

[33] Fung, C., Yi-Liang Chen, Xinyu Wang, Lee, J., Tarquini, R., Anderson, M., & Linger, R. oct 2005. Survivability Analysis of Distributed Systems Using Attack Tree Methodology. In *Military Communications Conference, MILCOM 2005. IEEE*, volume 5, 1–7, Seattle, WA. Network Centric Operations, Boeing Phantom Works.

[34] Michael Gegick & Laurie Williams. 2005. Matching attack patterns to security vulnerabilities in software-intensive system designs. In *SESS '05: Proceedings of the 2005 workshop on Software engineering for secure systems, building trustworthy applications*, 1–7, New York, NY, USA. ACM Press.

[35] Higuero, M., Unzilla, J., Jacob, E., Saiz, P., Aguado, M., & Luengo, D. oct 2005. Application of 'attack trees' in security analysis of digital contents e-commerce protocols with copyright protection. In *International Carnahan Conference on Security Technology, CCST '05. 39th Annual 2005*, 57–60. Fac. of Eng., Basque Country Univ., Bilbao, Spain.

[36] Dave Rayner. Survey of international activities in Internet-enabled metrology. Technical report, NPL, 2003.

[37] Pianegiani, F., Macii, D., & Carbone, P. jun 2003. An Open Distributed Measurement System Based on an Abstract Client-Server Architecture. *IEEE Trans. Instrum. Meas.*, 52, 686–692.

[38] Bertocco, M., Ferraris, F.and Offelli, C., & Parvis, M. 1998. An Client-Server Architecture for Distributed Measurement Systems. *IEEE Trans. Instrum. Meas.*, 47, 1143–1148.

[39] Winiecki, W. & Karkowski, M. 2002. A new Java-Based Software Environment for Distributed Measuring Systems Design. *IEEE Trans. Instrum. Meas*, 51, 1340–1346.

[40] Bertocco, M. & Parvis, M. 2000. Platform Independant Arcitecture for Distributed Measurement. *IMCT*, 2, 648–651.

[41] Michal, K. & Wieslaw W. May 2001. A New Java-Based Software Environment for Distributed Measurement Systems Designing. *IMTC*, 1, 397–402.

[42] Ewald, H. & Page, G. 2003. Client-Server and Gateway-systems for Remote Control. *IMCT*, 2, 1427–1430.

[43] Fjeldly, T. A. & Shur, M. S. 2003. *Lab on the Web, Running Real Electronics Experiments via the Internet*. Number ISBN: 0-471-41375-5. John Wiley & Sons.

[44] Dudley, R. A. & Ridler, N. feb 2003. Traceability via the Internet for Microwave Measurements Using Vector Network Analyzers. *IEEE Trans. Instrum. Meas*, 52, pp. 130–134.

[45] Ives, D., Parkin, G., Smith, J., Stevens, M., Taylor, J., & Wicks, M. Report to the National Measurement System Directorate, Department of Trade and Industry - Use of Internet by Calibration Services: Demonstration of Technology. Technical report, NPL, March 2004.

[46] Carullo, A., Parvis, M., & Vallan, A. 2002. Security Issues for Internet-Based Calibration Activities. In *Proceedings of the 19th IEEE Instrumentation and Measurement Technology Conference, 2002*, volume 1, 817–822.

[47] Barker, R. & Parkin Greame. Use of the Internet for Calibration Services Protecting the Data Final Report. CSMSE 28/03, NPL, Queens PRoad, Teddington, Middlesex, TW110LW, July 2003.

[48] Sand, A., Slinde, H., & Fjeldly, T. 2007. A Secure Approach to Distributed Internet-Enabled Metrology. *IEEE Trans. Instrum. Meas.*

[49] .NET. http://www.microsoft.com/net/basics.mspx. (Visited May 2007).

[50] GenuineChannels. http://www.genuinechannels.com. (Visited May 2007).

[51] RFC 2626 (HTTP 1.1). http://www.w3.org/Protocols/rfc2616/rfc2616.html. (Visited May 2007).

[52] Virtual Instrumentation Software Architecture. http://www.ni.com/visa/. (Visited May 2007).

[53] General Purpose Interface Bus. http://en.wikipedia.org/wiki/IEEE_488. (Visited May 2007).

[54] ITU-T. August 2005. X.509 Recommendation : Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks.

[55] The Transport Layer Security (TLS) Protocol Version 1.1. http://tools.ietf.org/html/rfc4346. (Visited May 2007).

[56] Wagner, D.and Schneier, B. November 1996. Analysis of the SSL 3.0 Protocol. In *The Second USENIX Workshop on Electronic Commerce Proceedings*, pp. 29–40. USENIX Press.

[57] Paulson, L. C. 1999. Inductive Analysis of the Internet Protocol TLS. *ACM Transactions on Information and System Security*, 2(ISSN 1094-9224), pp. 332–351.

[58] Zimmermann, H. Hubert 1980. OSI Reference Model-The IS0 Model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications*, 28(4), pp. 425 – 432.

[59] ISO/IEC. 2005. ISO/IEC 17799:2005, Information technology - Security techniques - Code of practice for information security management.

[60] August 1991. NS 5814 Requirements for risk analyses.

[61] Dotfuscator. http://preemptive.com/products/dotfuscator/index.html. (Visited May 2007).

[62] ISO/IEC. 2005. ISO/IEC 27001:2005 Information technology - Security techniques - Information security management systems - Requirements.