

VENetA

A secure network administration software with
customer connection

Anders Lilja
pettsson@cs.umu.se

May 8, 2006
Master's Thesis in Computing Science, 20 credits

Supervisor at CS-UmU:
Jonny Petterson (jonny@cs.umu.se)

Supervisor at Värnamo Energi AB:
Nicklas Lundin (nicklas.lundin@varnamoenergi.se)

Examiner:
Per Lindström (perl@cs.umu.se)

UMEÅ UNIVERSITY
DEPARTMENT OF COMPUTING SCIENCE
SE-901 87 UMEÅ
SWEDEN

Abstract

The metropolitan network in Värnamo is build and maintained by Värnamo Energi. The administration software that is in use today is old and incomplete. VENetA is a complete administration software designed for Värnamo Energi's network. It includes all required functions such as a customer database with contracts and services, network administration with statistics collection and surveillance and a matter handling system for both the staff and customer support. It also provides customers and tenants with a web interface to manage their own information and services.

The software was designed using a development process model that introduces risk analysis and security at an early stage. This helps developing a secure and stable software. In the process, risk analysis is a central part to describe the system environment and to make sure it is well protected.

Contents

1	Introduction	1
1.1	Problem description	1
1.2	Goal	1
1.3	Planning	2
1.4	Report contents	2
2	Security	3
2.1	Definitions	3
2.2	Threats	4
2.3	Policies and mechanisms	5
2.3.1	Access control mechanisms	6
2.4	Designing secure software	7
3	Risk analysis	9
3.1	Introduction	9
3.2	Methods	12
3.2.1	ISO/IEC 13335-3	12
3.2.2	NIST	13
3.2.3	CRAMM	14
3.2.4	ISRAM	15
3.2.5	Practical Threat Analysis	15
3.2.6	Microsoft Threat Modeling	16
3.3	Discussion	16
4	VENetA	19
4.1	Overview	19
4.2	Development process model	19
4.3	Methods	20
4.4	System environment	22
4.5	Requirements overview	22
4.6	Risk analysis overview	24

4.6.1	Method	24
4.6.2	Assets	24
4.6.3	Threats	24
4.6.4	Conclusion	27
4.7	Security policy	28
4.8	System design and prototype	31
4.8.1	Overview	31
4.8.2	Platform	32
4.8.3	Database	33
4.8.4	Security	35
4.8.5	Software	35
4.8.6	Restrictions	37
4.9	Other software	37
5	Conclusions	39
5.1	Future work	40
5.2	Reflections	41
6	Acknowledgements	43
	References	45
A	LAN topology	49
B	Database design	49
C	GUI	52
D	Usage scenarios	53
E	Functions description	57
E.1	GUI	57
E.2	DBI	58
E.3	Security	59
E.4	Network	60
E.5	System	61
E.6	Customer	62
E.7	Administration	63
E.8	Technical	63
E.9	Tasks	65
E.10	Statistics and surveillance	65

Chapter 1

Introduction

Värnamo Energi AB was founded in 1955 by a fusion of two power companies. It provides Värnamo and its surroundings with electric power, broadband Internet connections, cable television, district heating and other energy products. A couple of years ago, when the electric power market became unregulated, the company was split into two companies, one for the customer services and one for the power lines, Värnamo Elnät AB.

1.1 Problem description

The IT-department of Värnamo Energi AB has built up a communication fiber network from scratch. The process started less than 10 years ago and the network is still growing rapidly. Today there are about 800 homeowners, 750 tenants and 150 companies connected. A network of this size requires good administration software and the company has, until now, managed to use their own software. However, because of the rapid growth, the developer of today's system has no time to create a new system. The technicians have created add-ons to the first application, but these are still not synchronized because of technical differences.

Today's system is built up from several parts. The technicians have a separate system, which does not fully provide what is needed. The market department uses the original system, with a customer database and an incomplete technical database. There is also a documentation system for the fibers and cables, XOpto[34], which could be integrated with the administration software, but is not. The total system is definitely inefficient to work with and the lack of integration consumes much extra time for all parts.

1.2 Goal

The task of the project is to create a new administration software with a customer database including contracts and services, web interface for customers, network administration with SNMP (Simple Network Management Protocol) support, connection to the cable documenting system and a matter handling system; all included. The system must be secure since customers are allowed to access it.

Building such a system requires good documentation. The development process will

follow a software process model, which will be discussed in section 4.2. It should also include the development of a security policy and other security related design documents.

Building the entire system is not possible within the timeframe of this master thesis project. The primary goal will be to present a basic database and software design, with motivations of strategical decisions and a theory background of the security risk analysis. The secondary goal is to implement the system as an extension of this master thesis project.

1.3 Planning

The necessary steps to manage this project are described below.

- | | | |
|--------------------|---|---|
| Requirements | - | Gathering initial information. Finding the requirements of the system, together with the IT-personnel and writing a requirements specification. |
| Risk analysis | - | Finding the important assets for the company. Classifying assets in degree of importance. Investigating the threats to each asset and the existence of vulnerabilities. Gather it all together into a risk analysis report. |
| Security policy | - | Using the requirements and the risk analysis to establish an IT security policy for the software. |
| Database design | - | Using the requirements and the security policy to create a database design. |
| Tools selection | - | The tools to be used affect what can be done. Using the requirements and the database design, decide the tools to be used. This step should be conducted in cooperation with the staff. |
| Prototype | - | The user interface is important to make the users interested and to the success of the system. Therefore the prototype must illustrate how the system should look. |
| Application design | - | Using the requirements, database design and the prototype to design the application from the large parts down to each function. |

1.4 Report contents

This report begins with an introduction to security through the definitions of important concepts, as well as guidelines to create secure software. Chapter 3 will discuss the process of performing a risk analysis and compare different methods. The following chapter, 4, will discuss how the work was accomplished and present the results. Finally there will be a discussion of the results with a conclusion.

To fully understand this report, basic knowledge in computer networking and TCP/IP protocols might be necessary. Those who need to update their knowledge about these topics are recommended to do this before further reading, see [3]. If you need to read more about computer security, the following books are recommended; [2] and [1].

Chapter 2

Security

Security has become one of the most important parts of today's society. With more and more IT devices connected to each other, the community will be more and more dependent on it. With dependency comes vulnerability and to protect the resources, computer security is a must.

Most people use some kind of computer security almost every day; when paying the bills at the Internet bank, when using the credit card at a shop or in the cash dispenser or when shopping on the Internet. Some people also come in contact with sensitive data at work, which leads to usage of protection mechanisms such as cryptology and data access control.

2.1 Definitions

In the risk analysis sections there are many important concepts that must be understood. Here follows a few definitions of such concepts.

Confidentiality - This is the concealment of information and resources. It also applies to the existence of such data, because sometimes it can be just as important to know that there is data available as the data itself.[2]

Integrity - This refers to the trustworthiness of data and resources, and it is usually phrased in terms of preventing improper or unauthorized change. Integrity includes data integrity (the content) and origin integrity (the source of the data).[2]

Availability - This refers to the ability to use the information or resource desired. Unavailable resources might be at least as bad as nonexistent resources.[2]

Asset - Any kind of resource, physical or logical, that has any value to the organization.[8]

Impact - The damage or consequence of an unwanted action or incident affecting one or more assets. This can be loss of confidentiality, integrity or availability, damage, destruction, financial loss, etc.[8]

Threat - A potential violation of security. Threats can have both human and natural causes and can be accidental or deliberate.[8]

Vulnerability - A characteristic or property of an asset or group of assets which can be exploited by a threat to cause loss or damage. It can be an open way into the system, a weakness, a flaw or a backdoor. It can also be a human error.[8]

Attack - This comes up when a threat is realized. That is, an attacker uses a set of actions to perform a violation of security using a vulnerability in the system.

Risk - The potential that a given threat will exploit a vulnerability to cause an impact on the asset.[8]

Safeguards - An action taken or mechanism installed to reduce the risk; that is, limit the damage, eliminate the vulnerability and/or in any possible way reduce the impact of the attack. This may also be called *countermeasure* or *control*. [8]

Residual risk - The combined risk that remains after all possible safeguards has been implemented.[8]

2.2 Threats

There are different kinds of threats; some physical and some logical. The following types of threats applies to computer security; [2]

Burglary - Physical threats often consists of breaking into a closed area, accessing data and or physical machines. This causes physical loss and/or damage and may also harm information resources.

Nature - Disasters of different kinds may harm the system physically, causing financial, information and physical loss/damage.

Snooping - Unauthorized interception of information or resources. This is a passive attack, aimed at gathering information and is often used as a preparation step before an actual attack. An example of this is *wiretapping* when an attacker listens to the traffic on a network cable and tries to understand it.

Alteration - Unauthorized change of information. If the information is ordinary data, the attackers goal may be to make the modified data affect a decision for personal gain. If the data is program code, the attacker might get access to the machine. An example of such an attack is a man-in-the-middle attack, where the attacker has put himself in the middle of the communication between two parts, relaying information between them, possibly after modification.

Masquerading - The attacker tries to impersonate someone, without authority. A legal counterpart is *delegation*, where someone act on behalf of another, but not trying to be the other. The difference is important. A common use of this is to act as a manager and try to retrieve login information from employees.

Repudiation of origin - A false denial that an entity, either a person or a machine, has created or sent something.

Denial of receipt - A false denial that something has not arrived to the destination. Suppose the company that sent the bill state that they have not received any money, but the account owner has payed. Then, unless the account owner can prove that the company actually received the money, the account owner might need to pay again.

Denial of service - The attacker performs actions to prevent a service to function. This can be done by constantly, during a long period of time, increase the traffic to the server with huge quantities. An especially dangerous version of this attack is the DDOS, Distributed Denial of Service, attack where an attacker uses a large amount of hijacked computers on the Internet to create the huge load. It is very hard to predict and/or prevent such an attack.

2.3 Policies and mechanisms

Clearly, systems must be protected against threats. This requires a set of rules to describe how it should be done.[2]

Security policy - A document describing what is allowed and what is not, what must be done and sometimes how. A *user security policy* describes what users are allowed to do and not, while a specific *software security policy* might describe how a software should be used, by whom and in what way.

Security mechanism - A function that enforces some of the rules in the security policy. Security mechanisms aim to reduce the residual risk to an acceptable level pointed out by the security policy.

Naturally, not all rules can be enforced and therefore the security policy must be widely known by the employees. The following security mechanisms can be used to enforce parts of the policy.[2]

Password protection - Most systems require a username and a password to allow access to the system. This is a prevention mechanism, one that hinder the attacker from gaining access.

Cryptology - Data transfers and storage of sensitive data can use any kind of encryption to keep it secure. If a sufficiently good algorithm is used, then the only reasonable way to retrieve the data is to use the decryption key. Most attacks to such systems try to find out the key. If the algorithm is too weak, it could be possible to break the algorithm. This is also a prevention mechanism.

Access control - Access to data and services are often controlled by some kind of access control. Such services can be built into the operating system or directly in the application. It is a prevention mechanism that often is necessary to keep control.

Locks and burglary alarm - Physical locks can be used to stop physical access to computer systems. Unauthorized physical access is often fatal to a server system. The locks are often combined with burglary alarms, just to make sure that a break-in will be noticed. The lock is a prevention mechanism, while the burglary alarm is a detection mechanism.

Logs - A simple yet powerful detection mechanism. By logging users' logins, logouts, failed login attempts and abnormalities in the system, much can be seen when analyzing the logs.

Error correction - Some systems may have recovery mechanisms. Such systems are often combined with different kinds of detection mechanisms to find the error. When such an error (or attack) is found, it tries to recover from it. It can be file restoration from backup or similar techniques.

Mostly, several of these are combined into a system to protect it.

2.3.1 Access control mechanisms

As the system to be designed will include an access control mechanism, an introduction to such mechanisms is necessary. The most basic and complete method to describe access to objects is the *Access Control Matrix*, which is illustrated in table 2.1. The matrix keeps track of files and processes in a system and the processes access rights to each other. In the table, *process 1* may read from and write to *file 1*, but only read from *file 2*.

	file 1	file 2	process 1	process 2
process 1	read, write	read	read, write, execute	write
process 2	read	read	read	read, write, execute

Table 2.1: Access Control Matrix illustration

Using the Access Control Matrix directly to manage rights is inefficient. Since every process (including user processes) and file must be managed separately, the matrix grows very fast and quickly becomes impossible to overview.

There are primarily two types of access control; [2]

Mandatory Access Control (MAC) - A system controls the access to an object and the user cannot alter the access. This is also called *rule-based* access control.

Discretionary Access Control (DAC) - An individual user can set an access control mechanism to allow or deny access to an object. This is also called *identity-based* access control.

An operating system uses MAC, since neither the object nor the user can determine if access will be granted. The system uses a set of rules and information associated with both the object and the user to determine whether access is granted. As the system grows, defining access for specific users to specific object will be time consuming, if a direct Access Control Matrix is used. Therefore, many operating systems use groups or roles to assign access rights to groups of users. A group of users that is assigned specific rights allow all users in that group the same rights.

Another approach is the *role-based access control system*[2]. A role is a collection of job functions (that is a set of files and processes that can be accessed), such as system administrator, bookkeeper or shop assistant. To assign a user the necessary rights to perform the tasks of a shop assistant, the user need only be assigned the shop assistant role. This is also a MAC, but the access control is described using roles to which rules apply, instead of applying rules directly to users. These methods greatly simplifies maintainance of the system.

2.4 Designing secure software

It is important to bring security into the development process at an early stage. There are a lot of examples of weak software because security was inserted ad-hoc. By thinking-in security into the system from start, the complete system will not only be more secure, but probably more stable. The following guidelines are a good start when designing software.[4]

Secure the weakest link - No chain is stronger than its weakest link; a secure software must have a strong weakest link. An attacker will probably try to break the weakest link of the software and therefore it is always wise to start with the weakest links in the system. Performing a good risk analysis is fundamental to find such a link.

Practice defence in depth - A secure software should have more than one security system. If an attacker breaks the first level, there should be another protection mechanism. Such a design will protect the system from a full breakdown if an attacker manages to break the first mechanism. An example is an operating system; if an attacker manage break a user account, it might still only be able to access some parts of the system, to which the broken user account has access.

Fail securely - All systems fail sometimes, no matter how simple and secure they are. Therefore it is important that a failure is handled in a secure way, so that the system will not become vulnerable after the failure. Consider a system that, when fed with bad data, simply crashes and becomes unavailable. Such a system is good from the protection view, but very bad from the availability view. Therefore such a system is not reliable. Another system might leave the system running, but with an open trail into the system. This is also bad, since it gives the attacker a first step access to the system. So, when failing, make sure it fails securely.

Follow the principle of least privilege - Let the software or user have the minimum required access privileges to perform its necessary tasks. Doing so, the damage a system or a user can create is limited. Also, the access should be limited in time to only as long as necessary. A good example on Unix is that a software requires root privileges to open a TCP port below 1024. An email software listening to port 25 (SMTP) should therefore start with root privileges, acquire the port, return the privileges to the system and then continue its execution with restricted privileges. This will prevent the email software from being the bad guys tool.

Compartmentalize - Split up the system and privileges in parts. Access to one part will not imply access to another part. This way, a breach into one part will still not compromise the whole system.

Keep it simple - Try not to complicate things, complex systems are never easily understood. A complex system is more likely to contain bugs and other problems, it will be harder to maintain and analyse. A simple yet powerful system should be the optimal goal.

Promote privacy - Protect your users, systems and code. Do not provide any unnecessary information at login screens or other publicly available resources. There is probably no quicker way of loosing customers than to expose their personal information. But, remember the previously rule, users do not use systems that are complicated to use, therefore there is a tradeoff between privacy protection and simpleness.

Hiding secrets is hard - Security is a matter of secrets. Often, access to a system is granted using usernames and passwords and data is protected by an encryption key. To keep security high, such secrets must not be revealed. To prevent this from happening, it is wise to minimize the amount of secrets that must be kept. Try to use well known algorithms and well tested implementations of code, especially security code. Keeping the keys secret is hard enough, so don't use *my-own-very-special-encryption-algorithm* or similar stuff.

Be reluctant to trust - Trust no-one. It may seem like a cliché, but it is really true. If you trust some part, you probably extend your trust to all parts that the other part trusts. Before buying a security software, make sure you know the techniques behind it, so that you can decide if it is trustable or not. Some vendors selling security products tells you to trust them, because *they know what they are doing*. Without a motivation about why they should be trusted, they cannot be. To help identify such a software or software vendor, the *Snake Oil Warning Signs*[10] is a good place start.

Use community resources - Don't use home-brew security software or algorithms, trust the well proven, solid algorithms of the public community. As mentioned before, it is not the algorithm that should be secret, it's the key. Use the resources available; communities, mailing-lists and forums. Before using an algorithm in a security environment, make sure you understand it, or at least, that it is well-known to be good.

Combining this with common sense and a good book[4] is a very good start when designing secure software. When constructing security software though, further studies are recommended.

Chapter 3

Risk analysis

As stated in the previous chapter, network security has become an important topic nowadays. To be able to provide the necessary high security level, specialized methods are a must. This chapter will discuss the risk analysis parts of this process.

3.1 Introduction

Risk analysis is the process of identifying risks, their magnitude and identifying areas that need safeguards. A risk analysis should be performed relatively often, depending on the organization, the assets and the threat frequencies. An exposed organization such as a bank, a security firm or an Internet web hotel might need to perform risk analysis several times a year, while a less exposed company such as the local plumber or a grocery store might need to perform a risk analysis once every five years. A risk analysis should not take long time, a couple of days should be sufficient. It does not only consume time for the involved parts, the system or the environment might change if the process takes too long.

To perform a risk analysis, good knowledge about the system and its environment is necessary and there are many ways to achieve this knowledge, many different methods, software and good experts to ask. However, almost all methods have a basic methodology in common, which is presented here; [5]

1. Identify all assets and classify their importance.
2. Identify any threats, vulnerabilities or other issues to the assets.
3. Determine the risks and their magnitude.
4. Implement necessary safeguards.
5. Monitor the safeguards effect and schedule risk analysis.

The result of a risk analysis should be a document describing involved assets, risks associated with the assets and suitable safeguards that reduce the risks.

Assets

The earlier definition of an asset stated that an asset is anything with a value to the company. It can be physical resources such as buildings, cars, computers, networks, servers, locks, doors, products etc, or it can be logical resources such as digital data,

documents or software. It might also be other things such as customer relations and reputation.

Identifying assets is not a simple task. A good idea is to look at inventory lists and group the assets by type, then look at the difference between the real world and the lists. It might also be necessary to define a review boundary to narrow down the amount of assets. Double-check that no assets were missed during the inventory, since a missed asset in the risk analysis might cause severe loss. The next step is to agree upon a scale for valuation, which can be either a qualitative scale (very low - very high) or a quantitative scale (0 - 1 or the monetary value). This choice is often a matter of taste. The value of an asset can come from many sources; actual monetary value, emotional importance, expected impact of loss or damage, etc. The owner of the asset or the department using the asset should be responsible for the valuation.

Threats

The assets identified are all exposed to threats. A threat must not necessarily be disastrous, it might just be an indication that something can go wrong. Jenkins[14] identified two axioms for threats;

1. The same population of threats exists for all systems and networks.
2. The frequency of occurrence of a threat cannot be altered.

The first axiom states that the variety of threats that might occur is the same for all systems, only the likelihood of threat occurrence is different. The population of threats is infinite. The second axiom states that one cannot prevent a threat from occurring, but it is possible to prevent the success of the threat; that is the impact on the asset.

Given this, identifying threats is a matter of prior knowledge of the threat population. The important information about a threat is the occurrence frequency and the affected assets. In section 2.2 a few different threat types were listed for computer networks, but there are, as stated, infinite many. The best way to identify threats is once again to use a predefined list of common threats, either from a checklist or a software.

Vulnerabilities

The identified assets also have vulnerabilities. A vulnerability must not necessarily be a problem, that is, there exists vulnerabilities of assets for which no threat exists. A vulnerability cannot cause any harm itself, there must be a threat exploiting the vulnerability to create impact.

A good start when finding vulnerabilities is to examine the asset's properties and try to find ways to harm them. Search the assets environment to see if there are unauthorized ways to reach the asset, etc. As can be easily understood, there are infinitely many ways a vulnerability can exist. Therefore, it is good to use a predefined list to help searching for vulnerabilities, but remember to look beyond the list too.

Risks

With the assets, threats and vulnerabilities available, combine them into a list of risks. This often involves some kind of weighted calculation to receive a value describing each risk. This value can be either a numerical value or a qualitative notion (such as negligible, moderate or critical). The value must reflect the assets true value to the organization; that is, a combination of how expensive a successful attack may be and the possibility

that it does occur. This way, a very important asset which is unlikely to be attacked may result in a low risk value, while an asset with low value that is very likely to be attacked may result in a high risk value. Risks with a notion of *higher* risk should be considered necessary to minimize using some kind of safeguard, while risks with a notion of *lower* risk should be documented for future risk analysis.

Combining the different parts into a calculated risk value will require good knowledge about both security and the assets. Such knowledge is gained through experience and from experts or expert systems.

Safeguards

When a risk is too high, it must be reduced or handled in an appropriate way. The selection of a safeguard must respect both the risk level and the cost to implement it. For example, casting a computer into a block of concrete would most certainly prevent intrusion, especially since there are no cables connected; but this also makes the computer and its services unavailable, so this safeguard is not acceptable. Finding acceptable levels of risk and relevant safeguards may be difficult.

To find such safeguards, the starting point should be to examine the vulnerabilities and the threats that may affect them. A safeguard may use different ways to reduce the risk, according to the ISO 13335-3 standard[12];

- Avoiding the risk.
- Transfer the risk to another part, i.e. insurance.
- Reduce the threats.
- Reduce the vulnerabilities.
- Reduce impact.
- Detect upcoming impacts and recover.

The most important thing is to reduce the cost for the company, no matter if it is economical or emotional. If the asset is cheap, it might be cheaper to just buy a new, than to invest in protection mechanisms. There are software tools available even for this step. The outcome of this step is a list of the safeguards that are necessary to reduce the risk.

Management

When the risk analysis is finished, the safeguards are implemented and all runs smoothly, the results of the analysis should be verified. A new risk analysis will show if all risks are at an acceptable level, or if there are new risks, unsuccessful safeguards or missed threats/vulnerabilities/assets that need to be handled. Normally, the risk analysis ends with defining a security policy, an IT security architecture[12] or a similar document. This will tell the staff how often a risk analysis should be performed, what assets that need extra focus and how safeguards should be maintained.

3.2 Methods

The basic method described above may be used manually for smaller systems or companies, but as the organization grows, more effective methods will be necessary to handle the process within time. This section will shortly describe a few different models, but before this lets identify some important requirements of a security risk analysis method.[8]

Every risk analysis method should include;

common sense requirement - Some methods are very complex, yet they must all obey this requirement. No matter what valuation scale is used, there is a minimum and a maximum to that scale. When the asset has a high value, threats are likely to occur and vulnerabilities are easily exploited then the risk is high. When the asset has a low value, threats are unlikely to occur and no vulnerabilities that can be exploited exist then the risk is low. This implies that risk levels must be comparable, and also that countermeasures must be possible to rate.

business requirements - A risk analysis often results in safeguards that need to be implemented. A risk analysis method should therefore be able to find a balance between risk level and cost, since all safeguards cost to implement. To minimize cost, the method should be possible to use in any stage of a system life-cycle, because early detection of errors often yields cheaper correction. This implies that many users are involved in the process, including those that are not security experts, so care must be taken when formulating phrases. Finally, business risk level should be weight in for those assets that may affect the financial state.

functional requirements - The results of the risk analysis using a specific method should be deterministic and should not be affected by the users view. It should be possible to change detail level of the analysis, so that some parts can be focused on without digging into others. Several analyses using the same model should be possible to merge into a larger analysis. It should be easy and fast to use, but at the same time accurate. It should allow for statistics to contribute, as well as information from earlier analyses. Etc.

protection requirements - The risk analysis itself contains invaluable data to the organization. If such data should be contaminated, then the complete risk analysis is lost, and using the results can have catastrophic effects to the organization. Therefore, the risk analysis method should contain methods to protect itself, analyse its own data to find vulnerabilities and threats.

Most of the methods described in the following sections include the basic method described earlier in section 3.1. The first two sections below describe two standards, the ISO/IEC standard and the NIST standard. Then, a few other methods are described, including two software tools.

3.2.1 ISO/IEC 13335-3

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Committee) has created a standard for IT Security Management. The

ISO/IEC 17799[13] standard describes guidelines for practical IT security management. This standard refers to the Technical Report ISO/IEC 13335-3[12] for information about risk analysis methods.

This technical report describes the process of a security risk analysis and discuss different approaches.

The baseline approach - uses standard safeguards applied to the overall system. There are catalogues suggesting suitable safeguards for the most common threats. It is cost-effective for a small organization with low security needs, since it will require only a bare minimum of effort.

The informal approach - is based on prior knowledge, rather than on structured methods. This minimizes the effort needed, but there are many disadvantages; a degree of subjectivity will be introduced, the reviewers prior knowledge is directly affecting the results, it is easy to miss some assets or other details without a list, etc.

The detailed risk analysis approach - basically follows the method described in section 3.1. This requires a considerable amount of time, effort and expertise to succeed, but the results will reflect the actual situation and appropriate safeguards will be selected. This method is recommended for detailed systems, but not as an overall process.

The combined approach - combines the effect of the baseline and the detailed risk analysis approach. First, a rough risk analysis is performed to identify important IT systems. With the important systems, a detailed risk analysis is performed, and with all other systems, the baseline approach is used. This results in a acceptable overall risk level and even better where the detailed analysis was performed.

When the risk analysis is performed and appropriate safeguards are selected, the ISO/IEC 13335-3 also suggests a series of documents;

The IT System Security Policy - documents the required safeguards. The document describes how the safeguards work and why they are needed.

The IT Security Plan - describes how the security level will be maintained. It should describe the safeguards, how they are installed and maintained. It should also describe cost estimations for maintaining the security and, finally, the process of controlling that the safeguards actually were implemented correctly and work as expected.

Note that this standard does not state whether to use a quantitative or qualitative measurement of threats, vulnerabilities and risks.

3.2.2 NIST

The American NIST (National Institute of Standards and Technology) has released a Risk Management Guide for Information Technology Systems[20]. This document presents a method in 9 steps;

1. System characterization.

2. Threat identification.
3. Vulnerability identification.
4. Control analysis.
5. Likelihood determination.
6. Impact analysis.
7. Risk determination.
8. Control recommendations.
9. Results documentation.

The method described in this document is almost the same as the detailed risk analysis approach in the ISO document; that is the basic method described in section 3.1. It does not state whether a quantitative or qualitative measurement should be used.

3.2.3 CRAMM

1985, the UK Government tasked the Central Computer and Telecommunications Agency with investigating risk analysis and management methods[9]. This resulted in a new manual method called *CCTA Risk Analysis and Management Method*, CRAMM. This method has been updated several times and is now developed by Insight, Siemens, which recently released version 5.1. The method was initially developed for use within the UK Government departments, but is now a widely spread method also in the private sector.

The method contains three stages;

- Stage 1 - Find assets;
 - Define the review boundary.
 - Determine the value of data by interviewing users.
 - Identify and value physical assets.
 - Identify and value software assets.
- Stage 2 - Find risks;
 - Identify and assess threats.
 - Identify and assess vulnerabilities.
 - Combine threats and vulnerabilities to calculate the risk level.
- Stage 3 - Select safeguards;
 - From the risks in stage 2, select suitable safeguards to minimize the risk level.

As can be seen, the CRAMM method is quite similar to the basic method of section 3.1. The CRAMM software[29] is a good tool supporting this process. Actually, the software can be combined with a software called PRINCE (Projects IN a Controlled Environment)[9] which is a project management tool, to support software development projects using the SSADM (Structured System Analysis and Design Method) software development model. If used standalone, Insight states that CRAMM is better to use on an existing or nearly finished product, rather than during the initial phase of a software development. The CRAMM software primarily uses qualitative measurement, but, it can convert the results to quantitative values if needed.

3.2.4 ISRAM

The authors behind ISRAM (Information Security Risk Analysis Method)[7] refers to the two standards described above as "*robust and well-defined risk analysis methods*", but claims that following such a method may require the participation of a risk analysis expert to perform the risk analysis. They say "*ISRAM is designed for analyzing the risks at complex information systems by allowing the participation of managers and staff*". This means that the method should not require an expert and must be understandable for non-developers. ISRAM is designed as a quantitative, manual, risk analysis method that meet this requirement.

The method differs completely from the basic method described earlier in section 3.1. It is divided into seven steps, which are described here;

Step 1 - Determine if there is a security risk problem. Then divide the process into two parallel subprocesses, one that should determine *the probability of occurrence of security breach parameter*(P1) and the other *the consequences of occurrence of security breach parameter* (P2). The two subprocess methods are similar, so the method is described only once, for parameter Px.

Step 2 - List the factors that affect Px and weight those factors.

Step 3 - Convert the factors into survey questions, designate answer choices for the questions and assign numerical values to each choice.

Step 4 - Combine the survey questions and its answers into a risk table, which is a mapping between the results and numerical values.

Step 5 - Conduct the survey. The target group of the survey is all users of the system being reviewed, including ordinary staff.

Step 6 - Calculate parameter Px by mapping the answer of each question to the risk table. Combine weighted results of the questions into each parameter, then multiply P1 and P2 to obtain an overall result value.

Step 7 - Analyse the results of the survey and the overall value to find the risk level.

This method provides the risk level of a project, but it does not suggest any safeguards. However, the results from the survey can be used to estimate the risk level of different assets.

3.2.5 Practical Threat Analysis

PTA (Practical Threat Analysis)[11][32] is a software and method designed to help security consultants with the threat analysis of a software. It is a quantitative method, measuring risk level, potential impact and safeguard cost in dollars. PTA Technologies states that PTA is best used during the system design phase but, as threats, vulnerabilities and safeguards varies during the software lifecycle, threat analysis should be a continuous task.

In principle, PTA follows the basic method described earlier, but it does use some new terms to categorize assets, threats and vulnerabilities. The focus in PTA is the threats and how these can affect the assets.

3.2.6 Microsoft Threat Modeling

Microsoft takes another approach to threat modeling, the defensive approach, using their *Application Threat Modeling*[19] and the *Threat Analysis & Modeling* tool[22]. Their processes are focused on finding threats against an already designed application, from the defensive point of view.

The first step in the process is to model the software design into the software. This is done by following a set of design rules included with the software. Then the developers must find out what threats there are to the application, but not what the attacks would look like. After the threats are identified, the software tool maps the threats against an *Attack Library*, which includes definitions of known attacks. Just like an antivirus software, the *Attack Library* is frequently updated with new attacks, as such an attack becomes known.

Having identified the possible attacks, the software will point out specific vulnerabilities in the design, and then suggest appropriate safeguards, or countermeasures as they prefer to call it.

Microsoft has tried to create a threat analysis tool that can be used by developers and system designers, without being security experts, they claim.

3.3 Discussion

Most of the presented methods uses the basic method described in section 3.1 in some way. What differs between these methods is primarily how the method is used and what measurement scale that is used.

What is interesting, is that the standards reviewed, ISO/IEC and NIST, both look the same beyond the surface. The terminology differs a little, but both the ISO/IEC and NIST standard use the same basic method. What differs between the standards, is that the ISO/IEC standard defines four approaches, making it easier for a development team to perform a baseline risk analysis if the detailed risk analysis is not necessary. It also describes the combined approach, which takes the best out of both the baseline and the detailed risk analysis approach. No such thing is defined in the NIST standard. As expected, there are more standards available but these two seems to be the most important, since they belong to the American and the European (International) standardization organs.

The CRAMM method is interesting because the project started early, in 1985, so there are historical aspects. It also has another advantage, it is both a method and a tool, and it is also included in a complete project management software, making it easier to use.

ISRAM takes a totally different way when identifying threats and vulnerabilities; ask the users, they must know best. However, the survey manager must still identify the threats and vulnerabilities to be able to create the questions. I suggest a combination of the methods; identify the assets, threats and vulnerabilities, then conduct a survey to find the importance, the probability of an attack and the possible impact. This way, all users are a part of the risk analysis.

There are many tools available for risk analysis and management. But before selecting a suitable tool, it seems to me that a first, manual, overview risk analysis should be performed to see if a tool is really necessary. The tools (PTA and Microsoft Threat Modeling) seem quite complex to use and it is still necessary to identify and value the

assets. For a large project, using a tool might be necessary to succeed, but for a smaller project, a manual paper-based method might be just as good.

Finally, there are both quantitative and qualitative methods, even mixed methods, available. It seems to me that it cannot be concluded which is best; it depends on the analysis to be made. However, a qualitative method is harder to implement in software because of the subjectivity of the measurement, while a quantitative method seems hard and complex to use manually. Therefore, it seems that a qualitative method will be more suitable on smaller projects, while a quantitative method will most likely fit a large project better. It might also be important at which stage the project is; in the beginning it might be easier with a qualitative method due to unfinished documentation.

Chapter 4

VENetA

This chapter describes the VENetA software design, as well as the used development process model and the steps taken to create the design.

4.1 Overview

Today's system is split into two different softwares, excluding XOpto which will still be needed. The first software called *Stadsnätsadmin* was first created as a customer database, but was later extended to contain network administration routines. The database structure of this software does no longer match the network topology and the software is not well suited for network administration. The second software is a newly created matter handling system with network monitoring support. It includes network information, a customer database and is built as a customer support tool.

VENetA aims to be a combination of these two systems, plus a lot more. It includes a customer database including contracts and services, a network administration database including management of network equipment, statistics collection, surveillance and on-the-fly network status. It also provides a matter handling system, web interface for customers and staff and localization information for equipment and customers. VENetA should also be secure to use both from the internal as well as from the external network.

4.2 Development process model

This project is relatively small, so a simplified development process model is sufficient. The used development process model is based on the well-known spiral model[6] and is illustrated in figure 4.1. Black text describes steps actually done, dark grey text describes steps that was done indirectly, without being documented and light gray text describes necessary steps that needs be done in the future, before the software can be implemented.

The *Requirements plan* is the result of earlier work and is included into the *Requirements specification*, which describes the requirements of the system. The *Environment risk analysis* describes the systems environment and identifies the threats to a possible system location. The *Product prototype* illustrates a graphical user interface for the system. From the *Environment risk analysis* follows a *Security policy* describing what

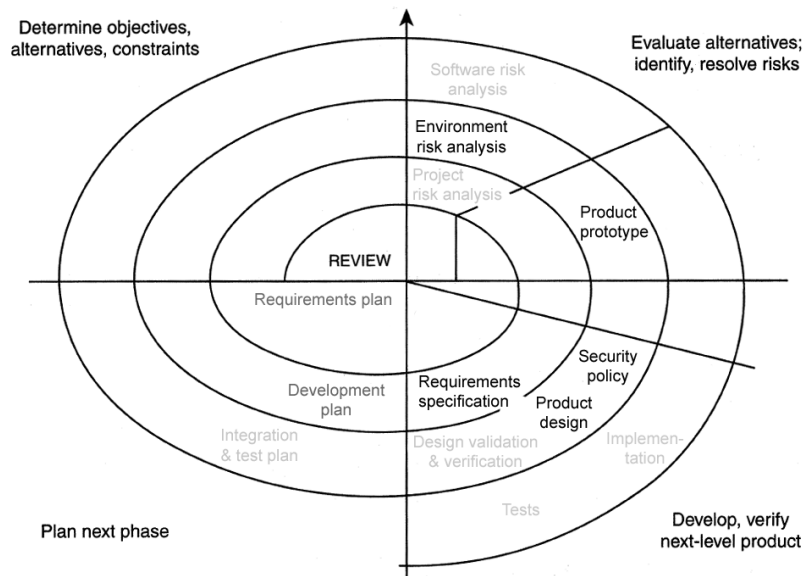


Figure 4.1: Used development process model.

is allowed and what is not in the systems environment including the entire workplace. Finally, a first *Product design* is created, describing how the system should work.

Following the development model, the design should be validated, tested and verified, then another risk analysis should be done before the system can be implemented. If any errors is detected during any phase, it is possible to take one step towards the middle of the spiral and then redo the tasks following the spiral from that point. As can be seen in the figure, only the first steps were taken during this master thesis.

4.3 Methods

The different phases include different ways to gather information and produce results. Below is a description for each step in the development process.

Requirements specification

The information needed to design the system was gathered from discussions with future users; that is primarily the technicians and the sales department. The results were documented in the requirements specification, see section 4.5.

Risk analysis

This will form the base for the security design of the system. Information for this part was gathered from several sources; the internal network structure, the metropolitan network structure, articles, books and from the Internet.

- The first step involved learning about risk analysis and finding a suitable model for it. The information for this step was found mostly in books and articles, but

some information came from the Internet.

- The second step involved examining the networks and the computer environment to find possible threats. Following this, the physical environment was investigated. The technicians assisted me with these steps by providing me with their knowledge, since there seems to be hardly any documentation at all. The only documents available are strictly technical notes in tabular format, unreadable for anyone except the invited personnel. Some information was collected using network examination tools such as a port scanner and a local network computer scanner.
- The third step was to use the information from the first two steps and try to find any exposed servers/services and examine the risks they were exposed to.

The results of this step is a risk analysis document describing the environment for the new system. An overview of the results can be found in section 4.6. Further risk analysis steps needs to be done after the design phase.

Security policy

Every company with a computer network should have a usage policy. To create such a policy, information about usage, resources and the employees is needed. It is also good to have seen a few such policies earlier.

Information about the workplace was gathered during the risk analysis. Complementary information about the employees behavior was collected using observation and discussions with a technician.

Information of what should be included in the policy was gathered primarily from the Internet, by reading several publicly available policies. Basic information was also gathered from books and articles.

The new security policy described in section 4.7 should replace the loose policy used today.

Prototype

In this step, prior knowledge about GUI (Graphical User Interface) design was used. Information about content and available functions was gathered by observing the usage of the different systems used today and by discussing the systems with the technicians. The prototype needs to be tested and verified before implementation.

Design

Using all previous documents and information from books, articles and prior knowledge, the design document was created. The following steps was needed to create the design;

1. The platform and tools to use when building the system were chosen. To do this, information about necessary tools and hardware platform was collected. In discussion with the technicians it was decided what tools to use.
2. The database was designed. The information to do this was gathered from prior work, todays system and a lot of discussions with the technicians.
3. The system was divided into parts based on the database design. The interaction between the different parts were investigated.

4. All parts and the interaction between them were documented. Functions for the GUI were added.
5. A few usage scenarios were created to test the design. A link-map of the system was created to support the usage scenarios and understanding of the system.

The results are documented in section 4.8

4.4 System environment

Figure A.1 illustrates the topology of the LAN (local area network) at Värnamo Energi, or at least the expected topology. The Internet connection consists of two links to different ISPs (Internet service provider) to ensure a connection is always available. These connects to the node VMO-OST, which is an extreme router that contains virtual routers. The MAN (metropolitan area network) is connected to a virtual router (VR Internet) which connects it to the Internet. To this virtual router, there is also a DMZ 1 network connected, containing servers provided for the customers of Värnamo Energi.

The company network of Värnamo Energi connects to the virtual router just like any other company. Internally, Värnamo Energi is protected from the MAN using a Cisco firewall, which separates the company's LAN from the public servers in DMZ 2. In the DMZ, there is a webserver, a DNS for the local network and a Citrix Secure Gateway. The level markers of the firewall describes where traffic is allowed; connections are always allowed from within a higher level to a lower, that is from within the LAN to the DMZ or the Virtual Router, or from the DMZ to the Virtual Router. All other connections are exceptions.

Router VE splits the internal LAN into four separate categories; the office network, the two networks of the Power department and the connection to WM-Data (for the customer billing system). Connected to this router is also the Administration Network of the MAN. This connection is a direct connection to a Virtual router (VR NetAdmin) of the VMO-OST node, that routes information to the network equipment controlling the MAN, using a separate VLAN address. A VLAN is a virtual LAN, that is it acts like a separate LAN, but uses the same physical cables.

In the office network, there are of course workstations, even though these are not included in the figure. Most of the users run terminal services, that is the Citrix framework, and the servers for this are the WTSx servers. There is also an SQL server, a domain controller DC1, an email server Exchange and three other resource servers.

4.5 Requirements overview

Figure 4.2 illustrates the necessary information and the connection between the information entities. Beginning with a customer; each customer that uses any kind of service must have an agreement, a contract. Bound to the contract are services (beginning with *srv*), which contain information about each service. Some services are connected to others, i.e. an email account may be connected to a domain. Other services require hardware connections to be established; these are bound to a connection. A connection is a way to describe the whole path from a central node to the customer, and it may be built up using links and VLANs. A VLAN is a virtual LAN, built up using links.

Above this, such a system requires a lot of support features. Below follows a summarized list of the most important requirements[15].

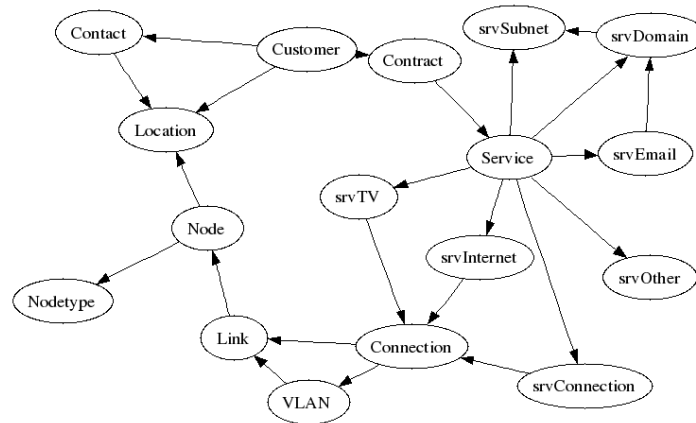


Figure 4.2: Necessary information entities in VENetA.

- *History* - The system needs to keep track of changes and an easy way to display a history list for important entities.
- *Customer interface* - Today's system allows for customers and tenants to manage their email accounts from the company's website. This must be possible or extended in the new system.
- *Editing* - It must be possible to add, edit and delete/hide all information entities.
- *Synchronization* - It should be possible to synchronize the customer database with the billing system KPlus[33], and the technical database with the physical map documentation software XOpto[34].
- *Technical* - It must support SNMP management of all equipment in the network and provide status and information from the network nodes.
- *Statistics* - The system must be able to collect statistics from all elements in the network. It should also allow for surveillance of services and nodes in the network, including an alarm system.
- *Services* - It should be relatively easy to add services and new functions to the system.
- *Security* - The system must be secured to prevent unauthorized access to information and network management. All users must identify themselves to the system.
- *Logging* - All changes to the system should be logged.
- *Tasks* - A matter handling system must be incorporated, allowing for documentation of support matters and similar.

4.6 Risk analysis overview

The importance of a risk analysis early in a development process cannot be stressed enough. This section describes the initial risk analysis for the project[16]. First, the used method is described and then an overview of the results from the risk analysis is presented.

Note - Some parts of the risk analysis is shortly described, leaving detailed information out on purpose. The information left out is considered a possible security risk for Värnamo Energi and its customers.

4.6.1 Method

The expected goal of the risk analysis performed at this early stage of the project was to examine the environment for the new system. If any vulnerabilities in the environment was found, suitable safeguards had to be identified. Therefore, the interesting results of the risk analysis is the list of threats, vulnerabilities and risks, as well as the appropriate safeguards.

As concluded in section 3.3, a qualitative method would be easier to use in a smaller project, and early in the process. This first step risk analysis was performed with just the requirements specification and knowledge about the environment.

Considering this, the method was chosen to follow the basic method described in 3.1, and the measurement to be purely qualitative. The degree of subjectivity introduced was accepted, since there must necessarily be another risk analysis before the implementation to find flaws in the design.

4.6.2 Assets

In the network described above, the most important assets is the VMO-OST router, the DNS servers, the DHCP server, the DC1 domain controller and the power supply to this equipment. This assets are important to keep the network and the network administration online. The software resources identified to be important is the DHCP service, the DNS services, the SMTP/POP/IMAP services for customer email, Exchange service for the staff email and the DC1 domain controller. Another important asset is the IT staff.

4.6.3 Threats

The following list describes some of the most important threats to the network, as well as some safeguards. The vulnerabilities are described indirectly. In the descriptions, *P* means *Probability of occurrence* and *IL* means *Impact Level*. The measurement scale for *P* is (lowest) *unlikely*, *possible*, *likely*, and (highest) *expected*. The measurement scale for *IL* is (lowest) *minimal*, *small*, *serious* and (highest) *critical*.

Unpredictable threats

Wind, Rain, Lightning

P: possible IL: small

Impact: physical damages

Description: cables and other equipment might be injured if exposed

Safeguard: protect the equipment by putting cables under ground and electronic equipment inside.

Fire, Burglary, Physical damage P: possible IL: critical

Impact: damaged or lost equipment

Description: impacts of this kind often means total loss of the equipment. This may also lead to information loss.

Safeguard: protect the equipment with locks, burglary and fire alarms where applicable and minimize access to the locations.

Illness, Accident P: likely IL: serious

Impact: loss of work and time

Description: in todays situation, an unexpected loss of one of the technicians for a long period of time would result in serious delays and economical loss.

Safeguard: Hire more staff. Share responsibility and schedules between the technicians. Make sure documentation is always up-to-date.

External threats

Trespassing towards the office network P: possible IL: serious

Impact: unauthorized access to data and resources

Description: a successful attack might lead to compromised data in databases and documents. It can also give access to the administration network, allowing for changes in the equipment.

Safeguard: close all possible ways into the office network. Make sure only authorized personnel have access to the network and establish a restrictive policy.

Trespassing towards the Exchange server P: likely IL: critical

Impact: loss of service, compromised emails and unauthorized access

Description: the exchange server is exposed to the Internet, since it allows retrieval of email messages directly. The server is located inside the office network, meaning that a successful hack of the exchange server could lead to access to the entire office network.

Safeguard: move the Exchange server to DMZ or install an SNMP-server-agent at a server in the DMZ, which then can deliver the email to the Exchange server using a secured connection.

Trespassing towards other parts P: likely IL: small

Impact: unauthorized access to data and resources

Description: a successful attack might lead to compromised settings in the network or the company's website. However, without access to the database in the office network, such errors should be easily corrected.

Safeguard: Prevent access to the administration network from outside the office network. Prevent access to DMZ servers from outside, except for the most necessary services.

Virus, Spyware

P: likely IL: serious

Impact: unauthorized access and compromised or destroyed data

Description: hostile code may destroy information and expose the network to unnecessary threats.

Safeguard: make sure protection software is installed and make sure the firewall(s) is correctly configured and effective.

Internal threats

System failures

P: likely IL: small

Impact: loss of resources and data

Description: if an harddrive crashes or a system goes down permanently, data could be lost and the work could stop.

Safeguard: make sure safe and accurate backup routines are available and documented. All IT staff should be able to perform a system backup and restore at any time.

Attempts to gain unauthorized access

P: minimal IL: serious

Impact: unauthorized access

Description: employees may try to gain unauthorized access, because of some kind of personal gain.

Safeguard: implement security mechanisms for sensitive data and resources, but most important, communicate the security policy to all the staff and make sure they understand and practice it

Use of unguarded computers

P: likely IL: serious

Impact: loss or manipulation of data and resources

Description: most employees at Värnamo Energi don't bother to lock their computer when leaving the room. The possible impact of an unauthorized usage of an unlocked computer depends on the logged in users rights and the unauthorized user's skills.

Safeguard: make sure all employees lock their computer when they leave the room. Also consider implementing automatically locking routines, even hardware based like smartcards. Communicate the policy.

Noticeable vulnerabilities

Below follows a short description of a few serious vulnerabilities of the network. These vulnerabilities should be taken care of as soon as possible, but are not easily solved.

Splitted administration network - VLAN is a relatively new feature of the IP networks. Värnamo Energi's network equipment consists of both older and newer equipment, where the older does not support the VLAN feature. The administration network VLAN is built up using free IP addresses in the 192.168.x.x series, but the network equipment without support for VLAN communicate via its public IP address. Knowing this, it can be concluded from figure A.1 that SNMP traffic will travel both the direct way through the *VR Netadmin* virtual router to the office network, and via the public *VR Internet* virtual router and the firewall to the office network. This way, it might be possible to attack a switch on the outside part of the administration network, and use this to reach the inner part and the office network. The best way to prevent this is to upgrade all the equipment, but there is not enough human resources for this time-consuming task.

SNMP-surveillance - The network equipment must always be running. To ensure this, a surveillance system must be available. This system must, because of the reasons described above, be able to reach both old and new equipment, which effectively forces the system to be run on a public server. This server is located outside of the Cisco firewall. Breaking this server yields access to the entire administration network, since it must have access to the administration network. To prevent this, the equipment must be upgraded to allow VLANs. Not until then, the system can be considered safe, since the surveillance system can be moved to the administration network without access to the Internet.

Social engineering risks - An effective method to break into a company's system is social engineering. This method is about using peoples credulity to retrieve information about the network, logins or even physical access via an employees computer or gaining access to keys and keycards. At Värnamo Energi, it would not be too hard for a socially skilled person to perform such an attack. To prevent this, the knowledge and acceptance of IT security must be considerably raised. There should be documented routines on how to sign up for a key or keycard.

4.6.4 Conclusion

There are several security issues in Värnamo Energi's LAN and most important of these are the risks of the splitted administration network and the Exchange server. It is therefore understandable that neither of the office network nor the VE DMZ can be considered as safe areas. There *are* safeguards protecting from most threats, but there are still obvious misconfigurations that should be prioritized.

Implementing the VENetA software is possible in any of the given locations, but it should be protected by its own safeguards. Access to the system and between system components should be encrypted using HTTPS/SSL connections and all required servers should run their own firewall and other necessary safeguards.

4.7 Security policy

The security policy[18] created is not a design document for the software, it is rather created as a direct effect of the risk analysis results and the loose IT security policy used today. The new policy is more suitable for an IT-intense company like an ISP and it is primarily a usage policy for the company. There are other policies for customers, included in their contracts.

Responsibility areas

The first part of the policy describes the scope of the policy, the responsibility areas and what will happen if an employee disrespect the policy. The responsible staff is named, so that every employee will know who to turn to if the policy is violated or with other technical issues regarding IT-security.

General guidelines

It is important that users know what is allowed and not, and the rules described in this section states that. When using the company's computer resources, the following rules must be obeyed;

- Each employee is personally responsible for using the resources in a correct manner.
- Each employee is personally responsible for any actions taken on a computer where the user is logged in.
- All activities on the company, including computer usage, must obey Swedish laws.
- Employees are allowed to use the computer resources for personal use after office hours.
- The following actions are *not* allowed;
 - Trying to gain access to resources without authorization.
 - Trying to hide one's identity or trying to act as someone else.
 - Trying to interrupt or break the intended usage of resources.
 - Trying to harm or destroy digital information.
 - Obviously waste available resources such as network capacity, staff time, office material, hardware or software.
 - Violate another employee's privacy life.
 - Use the company's resources to any kind of pornographic, racist or other offensive purposes.
 - Trying to interrupt any kind of safety software such as antivirus or firewall software.
- When using the WWW (World Wide Web);
 - Pages of suspectable characteristics should be avoided.
- When using email;
 - Emails may not contain any information with sensitive characteristics.

- Sensitive information may be sent via email if approved by the responsible director and the information is protected with encryption.
- Email resources is primarily for business duties.

Why? Following this rules will help protect information and resources, but also the staff's personal integrity and the company's economy. It may also minimize the amount of extra, unnecessary work.

Passwords

The following guidelines regarding passwords must be obeyed when using the computer resources, both internally and externally. Passwords should;

- Be hard to guess.
- Be at least 8 characters long.
- Contain a mix of lower and upper case characters, numbers and special characters.
- Not contain any words, names or common abbreviations from any common language.
- Not contain information that relates directly to the user, such as birthdate of children.
- Be changed within regular intervals.
- Not be one of the last 5 used passwords.

Tip. To help remembering the password, it can be created from a sentence. For example, *Are you sleeping, brother John? Morningbells are ringing.* Use the first letter of each word, any special chars and then add a number; lets say the product of the length of the first and last word. This yields: *Ays,bJ?Mar21.*

Why? Most common passwords contain words and are relatively short. There are software available today that can break such passwords in just a few minutes, some of them in just a few milliseconds.

Hardware

The following guidelines regarding hardware applies;

- Protect the equipment from physical damage, such as from water and heat.
- Servers and other central equipment not used by ordinary users must be placed in secured rooms, to which only authorized staff has access.
- Servers and other central equipment must be protected from power cuts using UPS.
- Consumed storage media must be erased or destroyed before disposal.

Software

The following guidelines regarding software applies;

- All software must be installed by system administrators.
- Antivirus and Antispyware software must be installed, updated and activated on the central server.

Also, for users with access to the local computer, the following applies;

- It is forbidden to install any form of server software such as a web server, ftp server or peer-to-peer file sharing software. If such a service is necessary, it should be installed by the system administrators.
- Antivirus and Antispyware software must be installed, updated and activated on the local computer.

Data resources

The company is dependent on the information stored in the system. Therefore, it is important that all staff only has access to it's corresponding documents and data. The following applies to documents and data resources;

- Every department is responsible for its related information resources and that information shared with other departments are correct.
- Documents created by staff at a department belongs to that department only, unless specified otherwise.
- Printed documents with sensitive information must be destroyed before disposal.
- Printed documents for archive should be stored in secured rooms, to which only authorized staff has access.
- Sensitive information should be treated with caution.

External connections

Any service that is exposed to the outside of Värnamo Energi office network must follow this guidelines.

- All communication must take place in a secure manner, preferably via encrypted connections.
- Access to the exposed services and the internal network must be as limited as possible.

The following applies for laptops and removable storage media;

- Removable storage media may ONLY be used where updated protection software is installed, such as Antivirus and Antispyware software.
- Laptops may be connected to the internal network if they are free from virus and other malicious code.

Routines

To ensure the system is maintained correctly, the administrators should;

- Regularly perform a risk analysis of the network and exposed resources, to minimize the amount of unnecessary vulnerabilities.
- Regularly check user accounts to make sure no inactive user accounts or services exists.
- Regularly perform backup of documents, databases and other information resources;
 - Weekly full backups to tape.
 - Daily backups to a disk array.
- Regularly check access logs to find possible break-in attempts and other threats.

4.8 System design and prototype

This section will present an overview of the VENetA design[17]. First, there are a few words that must be defined;

Tenant - A tenant is considered as a customer that uses services supplied by another customer to Värnamo Energi, without an agreement between the tenant and Värnamo Energi. Simply said, a tenant is a customers customer. But remember, a tenant can also be a customer with its own agreements with Värnamo Energi. A *landlord* is a customer that has tenants; in other words, the customer between the tenant and the company. A tenant can have more than one landlord if the tenant applies for services provided by more than one landlord.

Customer - A customer is a customer to the company, with its own contracts. However, in the design, the expression can be used interchangeably for a real customer and for an entity in the customer database table, which can be a customer, supplier or a tenant.

Contact - A contact is a physical person that can act on behalf of a customer. This is primarily useful for company customers.

Supplier - A supplier is a customer with one or more supplier contracts. A supplier contract means that Värnamo Energi buys a service from that company, in contradiction to all other contracts.

4.8.1 Overview

The basic structure of the system parts is shown in figure 4.3. All information the system needs is stored in the database, which is accessed via the DBI (DataBase Interface). The database interface classes consists of the main access class, the DBI, which contains functions for connecting to the database and handling queries. The DBI uses the help subsystems Network for SNMP access to the networking equipment and Security for checking permissions and handling access.

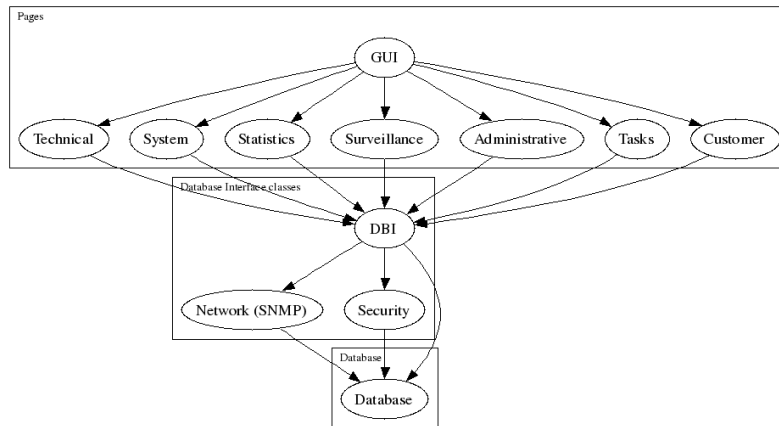


Figure 4.3: VENetA system overview.

The next layer of subsystems contains the real interaction with the user. The technical subsystem handles information about network equipment, links and connections. The System subsystem handles information about the VENetA users, called operators, and their access rights. Statistics can be collected for the network equipment, and the information about what should be collected and how to present it to the operator is managed by the Statistics subsystem. It is also possible to put surveillance of network equipment and servers, but also on information contained in the database. The surveillance subsystem takes care of this. Information about contracts and services is handled by the Administration subsystem, while customers and contacts is handled by the Customer subsystem. Finally, the Tasks subsystem is a task handling system. The GUI subsystem contains templates for displaying data and the graphical framework of the application.

4.8.2 Platform

Before entering the actual system design, it is a good idea to decide what hardware and software that is necessary. The system has a lot of possibilities, and the selected platform is definitely not the only possible choice. Below follows a description of the chosen platform, and why the specific tools was selected.

First, there are some basic requirements that must be addressed. In todays system, a statistics server software called RTG (Real Traffic Grabber) is collecting network information and is storing this into a MySQL database. This system part is a required component. RTG uses the Net-SNMP software library to access the network equipment. RTG runs on any POSIX compliant system, that is Linux, BSD and UNIX-clones.

Considering these requirements and the fact that the VENetA software should be based on web pages, the PHP framework should be used to develop the software. It is a good idea to minimize the amount of software installed on the server, both to save performance and to ease the installation and security configurations. RTG required a MySQL database and NetSNMP, so these can be used too. A Linux distribution is chosen as operating system because of prior knowledge and the RTG requirement.

This yields the following configuration;

Software	Version
OS	Linux (RedHat[28], SuSE[27], Gentoo[24] or equivalent)
Database	MySQL[21]
Webserver	Apache[23]
Application framework	PHP[25]
SNMP library	Net-SNMP[31]
Statistics	RTG[30]
Surveillance	Crontab or similar software

The hardware would preferable be a server machine, but a standard PC would perform just as well considering the small amount of users. Considering the relatively large database and that the statistics collection will run on the same computer, a minimum processor speed of about 1 GHz and at least 1 GB memory should be required.

4.8.3 Database

Appendix B illustrates the complete database design. The description is divided into sections, each representing a group of information resources. The sections is described below. Note that several tables have two important fields, *logop* and *timestamp*. These fields are used to log the action taken in the database. Tables without these fields either provide the functionality directly through other fields, or are considered not to need this logging.

Customer

This section contains two major tables, *customers* and *contacts*. The former of these describes customers, tenants and suppliers, while the latter describes contact persons of the customers. The *tenants* table describes the relation between two entities in the *customers* table, where one is the tenant and the other is the landlord.

Locations

In the system, *locations* are a way to organize physical locations into areas. The *areas* table describes a postal code area, that is all locations using the same postal code. The *streets* table contains information about street name and in which area the street belongs. Finally, the *locations* table describes a location on a street in an area. The location is represented not only with street address, but also with physical coordinates to be able to locate the location on a map.

Contracts

The *contracts* table describes the agreements made between a customer and Värnamo Energi. Such an agreement might be for a service supplied by Värnamo Energi to the customer, or it can be for a service supplied to Värnamo Energi by the customer. In the latter case, the agreement is a supplier contract. A contract can be a subcontract, for example if a contracts validity is extended, a subcontract can be added to notice this. In this case, the subcontract refers to the main contract via the *parentcontractid* field. This table also allows for physical documents to be scanned and stored directly in the database.

Network

This part contains network administration and statistics. The physical construction of the network is built up using links and nodes. The *nodes* table describes such nodes, and the *nodetypes* table describes the type of equipment of the nodes. Bound to the *nodetypes* table are a table describing the ports of the equipment and a table for SNMP commands, called mibs. Nodes can be of different type, it does not necessarily need to be a switch or a router. A node can be a computer, and it may have special services. The table *nodeservices* describes such services and the information can be used for surveillance. All nodes that allow SNMP access can be subject of statistics collection. The information necessary for this is stored in the *statistics* table.

The nodes are connected by links, and these are described in the *links* table. Links are bound together to form connections, which are dedicated paths through the network. Such a path may pass through both links and VLANs. A VLAN, in turn, also consists of links, even though it connects several nodes to each other forming a dedicated virtual network.

Services

Every service a customer uses must be bound to a contract. The services are documented in this section, including information about contracts. Some services require access to other entities such as connections, while others may depend on other services, such as the email service depends on the domain service. There is a special service called *srvconnection* which is a supplier service. Other communication companies can buy a so called *black fiber* which consists of only a fiber link, without equipment in the ends. Other services may be used by tenants, such as the Internet service. These services can be bound to a user customer directly via the *usedby* field, which means that one customer owns the service, while another uses it.

System

This section describes the access control system, which consists of the tables *operators*, *acesstable*, *roles* and *rights*. The access control system is described in section 4.8.4. Every operator is bound to a department within the *departments* table. The department information is primarily used in the matter handling system. The GUI is customizable for each operator, and this information will be stored in the *gui* table.

Included in this section is also the surveillance system, which is a very flexible surveillance system build around PHP-scripts included in the database. The tables included in this subsystem are *surveillance* and *survscripsts*.

Finally, there is a table called *sysinfo* which is a private system table that may contain information such as passwords and other important variables.

Matter handling

Every network administration software needs a matter handling system, and this is no exception. When a new matter is first created, information about the matter is stored in the *tasks* table. Following this, updates to the matter is stored in the *taskevents* table. There is also a private messaging system, which stores its messages in the *messages* table.

4.8.4 Security

VENetA has several different types of operators. There are employees which belong to different departments, there are customers, tenants and landlords. All these need separate functions and access. The access control system of VENetA is build on session control; every operator needs to log in. Once logged in, the information is retrieved from the database via the DBI subsystem. Access to an database entity is controlled before a question is asked, and if necessary, the query might be modified or denied.

Access to the system is controlled using *roles*. These roles have different access rights and they are very limited. To gain more access, an operator may be assigned several roles at the same time. The different access modes of a role is controlled via rights, which assign an entity with one or more access values; *read*, *edit* and *limited access*. The last value means that an operator with limited access may only access information related to itself, the customer it belongs to and possibly to the customers tenants.

Example - A tenant is logged in and tries to list all Internet services. The tenant has the access values *read+limited access* assigned to the Internet service table. Therefore, the tenant may only see its own Internet services, that is, where the tenant is assigned to the *usedby* field of the Internet service.

The entities that must be controlled are access to all database tables, the SNMP subsystem, the statistics subsystem and the VENetA system administration.

4.8.5 Software

PHP is a script-based language specifically developed for creating web-based solutions, but it is also object oriented, so classes can be used to create important routines. Normally, a PHP application consists of pages in combination with support classes. The first group in figure 4.3 is built primarily using pages and information formatting routines, while the lower rows is primarily built using classes.

Database Interface

The database interface subsystem shown in figure 4.3 has the responsibility to communicate with the database; all other subsystems must use this interface to communicate with the database or the SNMP module. When using the software, all information that is displayed on each page is collected from the database through a *SELECT* query to the database. When any data is written to the database through the *UPDATE* or *INSERT* commands, the DBI is responsible to add necessary information such as the logged in operator and a timestamp, as well as to filter the data and the query and decide if it is acceptable.

The following process describes how a *SELECT* query is handled;

1. The GUI creates the query on behalf of the operator.
2. The query is sent to the DBI.
3. The query is analyzed by the DBI according to contents and rights. These routines are in the security part of the DBI.
4. The resulting query is sent to the database, if accepted by the DBI.

5. The results of the query is received from the database and forwarded to the GUI for display.

The process is quite similar with the *UPDATE* and *INSERT* commands;

1. The GUI creates the command query on behalf of the operator.
2. The query is sent to the DBI.
3. The query is analyzed by the DBI according to contents and rights. These routines are in the security part of the DBI.
4. The query's content is analyzed and completed with necessary information.
5. The query is sent to the database.
6. No results is given from the database.

Statistics and surveillance

Two important subsystems are the statistics collection and the surveillance parts. The network statistics is collected using a software called RTG[30]. This software follows a script to collect the data, and this script is generated by the VENetA system. The necessary information to create this script is available in the statistics table in the database. When using the statistics data, a tool called *rtgplot* in the RTG software that can generate graphs on-the-fly will be used.

The surveillance part of VENetA is provided by means of PHP scripts. These scripts can be used to run checks on the database to find inconsistencies, human errors and similar things. It is also possible to check the network equipment configuration against the database using this method. The script environment is the PHP environment, but it should use the DBI to access the database, even though it is possible to use a direct connection to the database. The scripts are stored in the *survscripts* table and its runtime variables are stored in the *surveillance* table. Using this tables, a special script generates a *cron-job* script that is later executed by the machine's *crontab* software. The output from the scripts may be sent to the requesting operator that enabled the scripts, or the script may send it to any other operator or location.

GUI

Each operator may customize its working environment to suit their needs. This is done using favorites, color themes and similar methods and the customization data is stored in the *gui* table. The reason to separate this from the *operator* table is for security; every operator may customize its environment, and therefore requires write access to this table, but not all operators may change settings in the operators table.

A template for the GUI can be seen in appendix C. On top of the page there is room for some customizable shortcuts, and to the right there is a small box describing who is logged in, the operators department and actual date and time. The small fonted row below the shortcuts describes the actual page and its location in the navigation tree in figure D.1. The contents of the page is sectioned in three fields, two on the left side and one to the right. The first field to the left describes shortcuts to more information and subpages of the current page, while the second field contains current issues of the matter handling system. The field to the right contains information of the current entity, such as customer information. It also contains links to relevant subpages, such as contracts and services.

Functions

To fully understand the design, a navigation tree is present in figure D.1 in appendix D. This appendix also contains a few usage scenarios which illustrate common tasks. Appendix E contains a short description of every page and function in the system.

4.8.6 Restrictions

The design covers almost the entire system. However, there are functions that are not described in the system, which should be managed before the system is implemented.

- *Synchronization* with other important systems used at the company, XOpto and K-Plus. These systems also contain customer databases and related information, and XOpto contains map information about the fiber network. The information is stored in databases which could be examined to find corresponding information which can be synchronized.
- *Shortcuts* are not completely documented. The navigation tree in figure D.1 is not complete, it does only illustrate the basic structure. The missing links are mostly contents related and should be added during implementation.
- *Templates* to create contracts and services should be added. It is a very common task to create a contract with only one Internet service or TV service, so such templates should be available.
- The *GUI* is just roughly designed. The most important features are included in the *gui* class, but the design is not completely defined. To some extent, this is also an implementation issue.

4.9 Other software

There are similar systems available to buy, so this is not a *new* idea. The software with most in common with VENetA is NETAdmin[26]. It is a complete metropolitan network administration software featuring all of the requirements for VENetA, but it is rather expensive to buy and to use. In fact, the more customers, the higher cost.

Chapter 5

Conclusions

The primary goal of this thesis was to create a basis for a new network administration software for Värnamo Energi. The work has followed a software development model built on the spiral model, which introduces risk management early in the process. The following steps was performed during the work;

1. Requirements specification
2. Environment risk analysis
3. Security Policy
4. Software design

Requirements specification

The software used today includes many important functions, even though they are distributed over at least two applications. There are also new requirements that must be included. The system should include a customer database with geographic location information and contract handling, a network administration system with statistics collection and surveillance possibilities, an matter handling system and necessary security protection mechanisms. There is also a demand for a web-based interface available to customers.

Environment risk analysis

Before the risk analysis could be performed, information about possible methods and tools had to be gathered. Nearly all risk analysis methods follows a basic strategy;

1. Identify all assets and classify their importance.
2. Identify any threats, vulnerabilities or other issues to the assets.
3. Determine the risks and their magnitude.
4. Implement necessary safeguards.
5. Monitor the safeguards effect and schedule risk analysis.

This method is included in both the NIST standard[20] and the ISO/IEC standard[13] and technical document[12]. The goal of a risk analysis is to identify the risk levels; that is the combined value of the assets, the vulnerabilities, the threats, the likelihood that an attack occurs and the cost to repair or replace the asset when damaged. These values define where safeguards must be installed. The valuation scale of the assets, threats, vulnerabilities and risks may be either quantitative or qualitative, and the process may be manual or automated.

In the environment risk analysis, the basic method was used with a manual, purely qualitative, approach. The results of the analysis showed that there are several serious security issues in the systems environment, primarily on network protection level. Some of these could not easily be corrected because of the network equipment, while others were possible to handle with just reconfiguration of the network and the server software. The chosen method was easy to use, but definitely introduced a degree of subjectivity since all the valuations was made by me. This, however, was acceptable since there must necessarily be at least one more risk analysis before the implementation stage, to find flaws in the design.

Security policy

The risk analysis exposed the company's network and usage routines, which lead to a recreation of a usage security policy for the staff. This policy is primarily a step forward towards awareness of potential security risks in computers and computer networks. The most important parts describe what users are allowed to do and not, how passwords should be handled, how sensitive information should be handled and routines for backup and maintainance. It also specifies routines for environment risk analysis.

Software design

VENetA is a complete software for administration of a Metropolitan Area Network. It includes all important information assets; customers, contracts, services, network equipment and the links connecting the nodes into a network. It does also include a matter handling system with a strong connection to the other assets. The GUI of VENetA is divided into three different interfaces; one for staff, one for tenants and one for customers. The system is web-based and therefore available from almost everywhere. The system is protected with an internal role-based access control mechanism.

5.1 Future work

The system can definitely be implemented based on this thesis. There are, however, a lot of work left to do before an implementation can begin;

- The design must be evaluated and tested.
- A new (software) risk analysis must be performed to find flaws in the design.
- A more detailed prototype should be created and evaluated.

After this, the implementation and testing phase may begin. To use the software, some kind of database conversion tool must be created to transform all database information available today into the new system. It is also necessary to define the roles and the access controls more precise; that is, a software security policy should be created.

5.2 Reflections

The work was more time-consuming than expected so some parts had to be removed from the project. My first plan was to implement the system roughly, but there was no time left when I finally finished the design document. The resulting design is definitely good enough for a future implementation; I am satisfied with the work done so far.

Chapter 6

Acknowledgements

Special thanks to my internal supervisor, *Jonny Pettersson*, for fast and constructive response during this thesis. I would also like to thank *Roger Garpenholm* and *Richard Wennerström* at *Värnamo Energi AB*, who has provided me with a lot of information and lend me their time for discussions. Thanks also to *SiS* and *Måns Diedrichs* who kindly sponsored me with the *ISO/IEC 13335-3* and *ISO/IEC 17799:2005* standards, which otherwise are rather expensive for a student.

Finally, I would like to thank my wife *Jenny* for supporting me during the project, and my daughter *Amanda* who was born in February 2006, for changing my point of view in life.

References

Books

- [1] S. Bhasin. *Web Security Basics*. Course Technology, (Ebrary database), 2002. ISBN: 1-59200-006-1.
- [2] M. Bishop. *Computer Security, Art and Science*. Pearson Higher Education, 2002-12. ISBN: 0201440997.
- [3] K. W. Ross J. F. Kurose. *Computer Networking; A top-down approach featuring the Internet*. Pearson Education, Inc., 2003. ISBN: 0-321-17644-8.
- [4] G. McGraw J. Viega. *Building Secure Software*. Addison-Wesley, 2005-06. ISBN: 0-201-72152-X.
- [5] Thomas R. Peltier. *Information Security Risk Analysis*. CRC Press LLC, (Ebrary database), 2001. ISBN: 0-8493-0880-1 (e-book 0-203-99750-6).
- [6] S. L. Pfleeger. *Software Engineering, Theory and Practice*. Prentice Hall, Inc., 1998. ISBN: 0-13-624842-X.

Articles

- [7] I. Sogukpinar B. Karabacak. Isram: information security risk analysis method. *Computers & Security*, (24):147–159, 2005.
(<http://dx.doi.org/10.1016/j.cose.2004.07.004>, available 2006-04-19).
- [8] Z. Ciechanowicz. Risk analysis: requirements, conflicts and problems. *Computers & Security*, 16(3):223–232, 1997.
([http://dx.doi.org/10.1016/S0167-4048\(97\)00004-7](http://dx.doi.org/10.1016/S0167-4048(97)00004-7), available 2006-04-19).
- [9] Insight Consulting. Integrating security into it projects and programmes. *website technical paper*, 2005-10.
<http://www.cramm.com/files/techpapers/Integrating0Projects>
- [10] Matt Curtin. Snake oil warning signs: Encryption software to avoid. *Interhack.net*, 1998-04.
<http://www.interhack.net/people/cmcurtin/snake-oil-faq.html>, available 2006-04-19.
- [11] Ygor Goldberg. Practical threat analysis (pta) for the software industry. <http://www.ptatechnologies.com/>, 2005-01-10.
<http://www.securitydocs.com/library/2848>, available 2006-04-19.

- [12] ISO/IEC. Information technology - guidelines for the management of it security, part 3. *ISO/IEC TR 13335-3*, 1998-06.
Provided by SiS for this thesis.
- [13] ISO/IEC. Information technology - security techniques - code of practice for information security management, 2nd edition. *SS-ISO/IEC 17799*, 2005-08.
Provided by SiS for this thesis.
- [14] B. D. Jenkins. Security risk analysis and management. *Whitepaper, Countermeasures Inc.*, 1998.
http://www.nr.no/~abie/RA_by_Jenkins.pdf, available 2006-04-19.
- [15] Anders Lilja. Veneta kravspecifikation. Included work (Swedish), contact for a copy.
- [16] Anders Lilja. Veneta riskanalys. Included work (Swedish), contact for a copy.
- [17] Anders Lilja. Veneta system design. Included work (Swedish), contact for a copy.
- [18] Anders Lilja. Veneta säkerhetspolicy värnarn energi. Included work (Swedish), contact for a copy.
- [19] M. Talhah Microsoft Corporation. Microsoft application threat modeling blog. *MSDN Blog*, 2006-03.
<http://blogs.msdn.com/threatmodeling/archive/2006/03/16/553232.aspx>, available 2006-04-19.
- [20] NIST. Risk management guide for information technology systems. *NIST Special Publication 800-30*, 2002-06.
(<http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf>, available 2006-04-19).

Software

- [21] MySQL AB. Mysql 5.0 community edition.
<http://www.mysql.se/downloads/mysql/5.0.html>, available 2006-04-19.
- [22] Microsoft Corporation. Microsoft threat analysis & modeling v2.0 beta2.
<http://msdn.microsoft.com/security/securecode/threatmodeling/acetm/>, available 2006-04-19.
- [23] The Apache Software Foundation. Apache http server.
<http://httpd.apache.org/>, available 2006-04-19.
- [24] Inc. Gentoo Foundation. Gentoo linux.
<http://www.gentoo.org/>, available 2006-04-19.
- [25] The PHP Group. Php, hypertext preprocessor.
<http://www.php.net/>, available 2006-04-19.
- [26] Netadmin System i Sverige AB. Netadmin.
<http://www.netadmin.se>, available 2006-04-19.
- [27] Novell Inc. Suse linux enterprise server 9 (10).
<http://www.novell.com/products/linuxenterpriseserver/>, available 2006-04-19.

-
- [28] RedHat Inc. Redhat enterprise linux.
<http://www.redhat.com/en.us/USA/rhel/>, available 2006-04-19.
- [29] Siemens Plc / Insight. Cramm expert/express.
<http://www.cramm.com>, available 2006-04-19.
- [30] A sourceforge project. See homepage. Rtg - real traffic grabber.
<http://rtg.sourceforge.net/>, available 2006-04-19.
- [31] A sourceforge project. Various authors. See homepage. Net-snmp.
<http://net-snmp.sourceforge.net/>, available 2006-04-19.
- [32] PTA Technologies. Pta - practical threat analysis.
<http://www.ptatechnologies.com/>, available 2006-04-19.
- [33] WM-data. K-plus, a part of lettera.
<http://www.wmdata.se/wmwebb/menu1/branches/showBranchLevel3.asp?TId=0&BIId=0&B3Id=185>, available 2006-05-03.
- [34] X-Opto. Oaktree software ab.
<http://www.oaktree-sw.com/OaktreeSWE/defaultSWE.html>,
available 2006-04-19.

A LAN topology

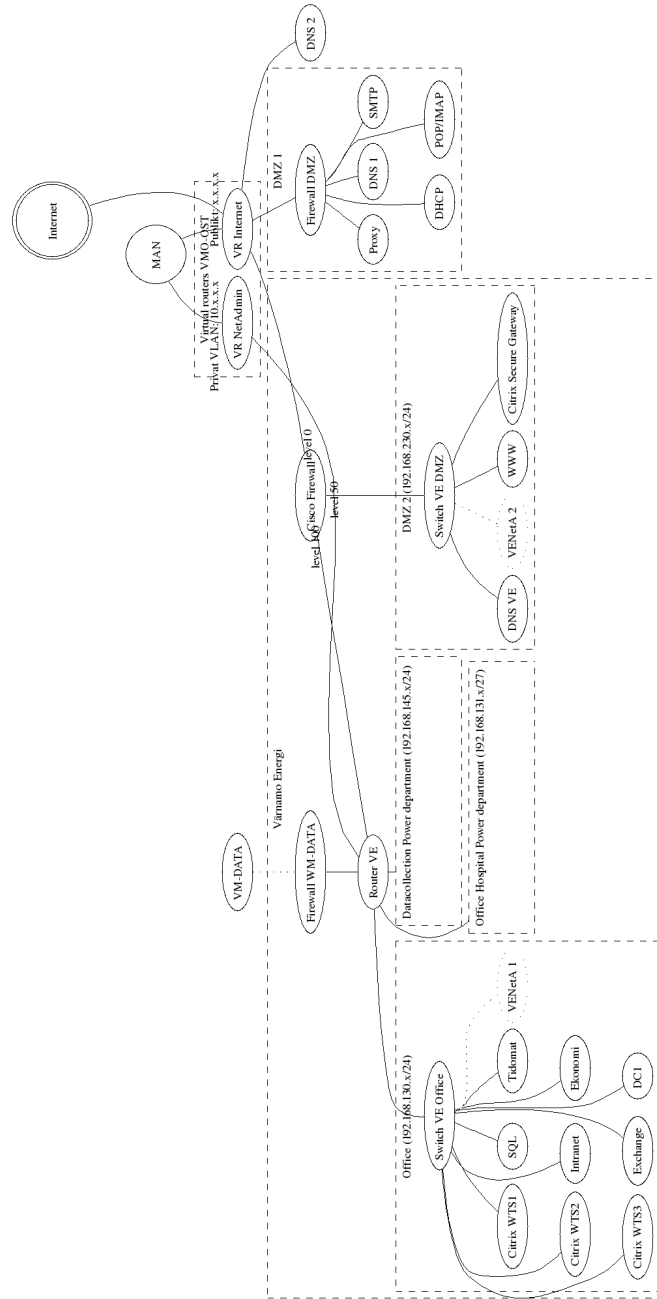
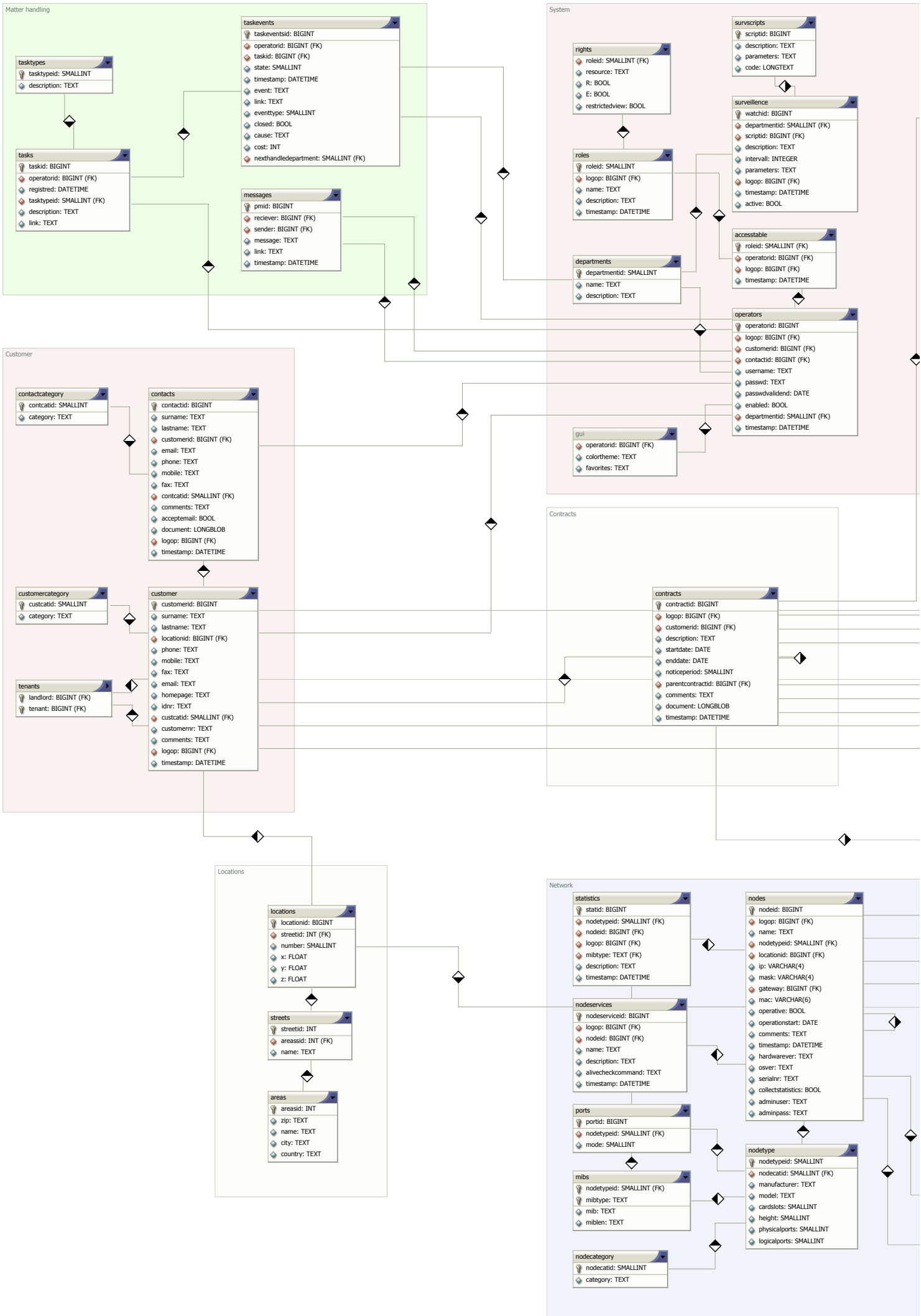


Figure A.1: Värnamo Energi LAN topology



Startpage

Favorite 1

Favorite 2

Favorite 3

Favorite 4

Operator: <logged in>
Department: <dept>
YYYY-MM-DD HH:MM

Start > Subpage > Subpage > ...

Header

More information

Shortcut 1

Shortcut 2

Shortcut 3

Shortcut 4

...

Current issues

YYYY-MM-DD

Issue 1

YYYY-MM-DD

Issue 2

YYYY-MM-DD

Issue 3

More issues...

Contents depending on page

D Usage scenarios

Here follows a few usage scenarios to illustrate the work flow. The scenarios are built on the navigation tree in figure D.1.

Scenario 1 - Tenant

A tenant want another email account, but does not remember the password to the website.

The tenant enters the website where he is presented the *login.php* page. There, he follows the link *forgot password* and is transferred to the *changePASS.php* page. Here, the tenant enters his username and/or personal ID number to request the new password. The company informs the tenant about the new generated password via email or ordinary mail.

With the new password received, the tenant revisits the website to login with the new password. Having done this, the tenant is transferred to the *tenantindex.php* page. Then he follows the *add services* link to move to the *services.php* page. He choose to add an email account and moves on the *srvemail.php* page. There, he enters the new email account name and confirms. The email account is checked immediately and the tenant will receive the confirmation, if the account is accepted. He is finished and logs out.

Scenario 2 - Customer

A customer wants to manage its email services and look at its tenants list.

The customer logs on to the website and receives the *customerindex.php* page. Here, the customer can navigate to *tenantslist.php* to see the tenants list and further on to *editcustomer.php* to find more information about a tenant. The customer can also navigate from the startpage to *contractlist.php* to list the customers contracts. Within each contract, there is a list of services, presented to the user at the *contract.php* page. The customer manages the email accounts in the different contracts and then logs out.

Scenario 3 - Technical

An operator at the IT department wants to add a new switch of a new type to the network.

The operator logs in and receives the *operatorindex.php* page. From the startpage, the operator navigates to the technical pages by following the link to the *technical.php* page. Because the switch is of a new type, the type must first be added. He navigates to the *nodetypes.php* page and follows the link to add a new nodetype. He ends up on the *nodetype.php* page where he enters the information about the equipment, including the SNMP MIB commands. Then he saves the information and follows the direct link to add a new switch of this nodetype. He is transferred to the *node.php* page, where he enters the information about the node. He saves the information and navigates to one of the subpages to add statistics information and/or special node services.



Figure D.1: Navigation tree of VENetA

Scenario 4 - Customer support

The customer service expedition receives an error report and wants to add it to the matter handling subsystem.

The operator logs in and navigates from the startpage via *customer.php* to the *customerlist.php* page. He uses the customer search function on this page to locate the calling customer and selects it. Then he chooses to add a new event and is transferred to the *task.php* page. The operator enters information about the error; the error type, the department to handle it and a description. He may also add a link list to related entities such as an Internet service or the customer. Technically, the link list may contain any entity from the database. Finally, the new matter is saved.

Scenario 5 - Customer technical support

A technical operator wants to follow up a matter about requested upgrade of an Internet service from 2 mbit to 10 mbit. The issue also contains troubleshooting of a sudden breakdown of the existing Internet service.

A customer has requested an upgrade of his Internet service. The market department has accepted the upgrade and forwarded the issue to the IT department for realization of the upgrade. During the process, the customer calls technical support to report an error in the not yet upgraded Internet service.

First, the Internet service should be working before an upgrade should be realized. The operator logs in and navigates via the startpage's issue list directly to the actual matter. The operator then follows a link in the linklist to the actual Internet service, described at the *svvinternet.php* page. On this page, the operator navigates to the connection page and receives information about the link status on the entire connection from the customer to the networks Internet access point. The switch closest to the customer reports that the port is closed. A presumed mistake makes the switch configuration inconsistent with the database and the operator activates the port. The operator navigates back to the Internet service page and changes the service capacity to 10 mbit. When saving the change, the system automatically reconfigures the switch to the correct speed.

Scenario 6 - Administration

A new home owner orders a TV service and a 10 mbit Internet service.

An operator from the market department logs in. From the startpage, he navigates to the *customer.php* page and selects new customer. On the *editcustomer.php* page, he enters the personal contact information of the new customer and saves the information. Then, he follows the direct link to add a contract for the customer. The standard contracts for these services contains only one service, and therefore the operator must create one contract for each service. He enters the information necessary for an Internet service contract and saves the contract. Then he adds an Internet service using the add service function and is transferred to the *svvinternet.php* page. There, he enters the information for the Internet service, including capacity. The customer lives in a detached house where network equipment is already in-

stalled, so there is already a connection. The operator locates the correct connection, binds it to the Internet service and saves the new service. The operator then returns to the customer page and performs the same steps for the TV service; add a new contract and then add the service to the contract. The TV service uses the same connection as the Internet service, even though it is different fibers. The TV service is saved.

E Functions description

The file structure of the application is build according to figure 4.3. First, the start-pages is present in the root folder, then each subsystem has its own subfolder.

For all accessible pages, the access rights is always controlled before any information is displayed. For example, if an operator do not have the roles necessary to add a new role, he is prevented to do so.

In the descriptions below, the parameters describes public parameters, that is, parameters that may be defined by other pages to control the information on the page. A page may also define private parameters that is only used internally when reloading the same page with request for new information; such parameters may be added during the implementation.

E.1 GUI

index.php

This is the main start page. From this page, the login process and forwarding to the other start-pages are performed.

tenantindex.php

Startpage for tenants.

customerindex.php

Startpage for customers and suppliers.

operatorindex.php

Startpage for employees and other staff.

format.php

This page is actually a preprocessed CSS stylesheet file that controls the system appearance.

class GUI

This class defines base functions for the user interface. Standard functions to create tables, building menus, displaying shortcuts and similar are defined here.

gui.php GUI.headermenu

Generates html code for the shortcuts menu on top of each page, with respect to the current operator and page.

gui.php GUI.infomenu
Generates html code for the upper left box, with respect to the current operator and page.

gui.php GUI.tasklist
Generates html code for the actual matter list, with respect to the current operator.

gui.php GUI.displaytable
Generates html code for a table containing data from an array.
Parameter *header* - An array of size Nx1 with headers.
Parameter *content* - An array of size NxM with M rows of information for the table.

gui.php GUI.infofield
Generates html code for a floating information field.
Parameter *content* - The contents of the information field. The parameter may contain a string or an array of strings.

E.2 DBI

class DBI

dbi.php DBI.connect
Connects the DBI to the database. This must be called for every time a page is loaded, but the routine should connect to the database when a new session is established and then keep the connection alive.

dbi.php DBI.disconnect
Disconnects from the database. This is normally not needed, because it is automatically done when a session terminates.

dbi.php DBI.query
Runs a query against the database. A query might be either a *SELECT*, *UPDATE* or an *INSERT* command. For many tables, the information is not updated, but instead a new row with the new information is added.
Parameter *query* - A string containing the query.
Return - The result of the query if FALSE in case anything went wrong. For a *SELECT* query, the results is data which is returned in form of a NxM array, where the first row contains the column headers. For the other query types, the result is TRUE for a successful update. An error description may be returned using a global variable (similar to *errno* in C).

dbi.php DBI.resolvlink

A few tables contains the field *link*, which includes references to related information entities. This function resolves such links into an array of html code, one for each link.

- Parameter *linklist* - A comma separated list of name-value pairs according to *table name = id*, each describing a link.
- Return - An array of html code strings, one for each name-value pair that could be resolved.

E.3 Security

login.php

Displays a login page and manages the login. It requests the Sec.login function to verify the login and establish a session.

- Return - Session variables or nothing.

changepass.php

For logged in operators, this supplies a method to change or generate a new password, according to the IT security policy that applies. For others, this supplies a method to generate a new password to be sent to the operator in a suitable way.

class Sec

This class manages login and passwords. Note that this class contains routines that must be called from every page that is included in the system, to maintain the security.

sec.php Sec.verifylogin

Verifies that a user session is established and that the login was correct. If not, the routine immediately redirects the user to the login page. If it was correct, the routine returns silently.

- Return - True at a valid session, false otherwise.

sec.php Sec.login

Verifies a login. At a successful login, a session is established and session variables are stored that can be verified by the verifylogin function.

- Parameter *user* - Username
- Parameter *pass* - Password
- Return - Session variables or nothing.

sec.php Sec.logout

Closes a session and removes session variables. This is automatically performed when the browser is closed, but for security reasons, a logout function is provided. A neutral page should be presented after logout.

sec.php Sec.hashpasswd

Calculates a hash from a password. This is the value stored in the database and is used when verifying the login.

- Parameter *pass* - The password to be hashed
 Return - The resulting hashed password in a string.

sec.php Sec.changePASS

Supplies the possibility to change password. This is an alternative to the *generatePASS* function.

- Parameter *user* - The user who wants to change password.
 Parameter *oldPASS* - The old password
 Parameter *newPASS* - The new password
 Return - True if the change succeeded, false otherwise.

sec.php Sec.generatePASS

Supplies the possibility to generate a new password according to the current security policy. This is an alternative to *changePASS*, though this function requires system access.

- Parameter *user* - The user who wants to generate a new password.
 Return - The new password.

class DBSec

This class contains functions to enforce the access control against the database.

dbsec.php DBSec.checkSQL

This routine verifies a query against the database and the operator's access roles. All routines must use this method before accessing the database. It uses session variables to verify the operator's access.

- Parameter *query* - The requested query.
 Return - True if the query was accepted, false otherwise.

E.4 Network**class SNMP**

This class contains functions for SNMP access. The functions uses the database to retrieve SNMP commands and other necessary information.

snmp.php SNMP.VEGet

Synonymous to SNMP Get, which retrieves a single value.

- Parameter *node* - The target node; that is a switch or similar equipment, identified by node id.
- Parameter *command* - The command to perform. This is defined with *PortStatus*, *PortSpeed*, *PortMac* or a similar command name. The correct MIB is retrieved from the database.
- Parameter *item* - The id of the information object requested, for example a port number.

snmp.php SNMP.VEWalk

Synonymous to SNMP Walk, which retrieves a series of values at the same command.

- Parameter *node* - The target node; that is a switch or similar equipment, identified by node id.
- Parameter *command* - The command to perform. This is defined with *PortStatus*, *PortSpeed*, *PortMac* or a similar command name. The correct MIB is retrieved from the database.

snmp.php SNMP.VESet

Synonymous to SNMP Set, which changes a single value.

- Parameter *node* - The target node; that is a switch or similar equipment, identified by node id.
- Parameter *command* - The command to perform. This is defined with *PortStatus*, *PortSpeed*, *PortMac* or a similar command name. The correct MIB is retrieved from the database.
- Parameter *item* - The id of the information object to be changed, for example a port number.
- Parameter *value* - The new value of the object.

E.5 System

system.php

Startpage for the *System* subsystem

oplist.php

Displays a list of operators. It should be possible to select an operator to edit.

editop.php

Display and edit the information of an operator.

- Parameter *id* - Operator id. If missing, create a blank form for a new operator.

rolelist.php

Displays a list of roles. It should be possible to select a role to edit.

editrole.php

Display and edit the information for a role.

Parameter *id* - Role id. If missing, create a new role.

departments.php

Display and manage departments.

E.6 Customer

customerlist.php

Displays a list of customers, and their relation to other customers.

editcustomer.php

Display and edit a customer, including relations to other customers and contacts. Displays an overview of the customers contracts, services and tenants.

Parameter *customerid* - Customer id. If not defined, create a new customer.

contactlist.php

Displays a list of contacts, and their related customer.

editcontact.php

Display and edit a contact.

Parameter *contactid* - Contact id. If not defined, create a new contact.

Parameter *customerid* - The customer id to bind a new contact to.

tenantslist.php

Displays a list of tenants and landlords. To change this information, look at the tenants customer page.

Parameter *filterid* - Filters the list on landlords so that only tenants to a specific landlord is displayed. Defined by the landlords customer id.

locationlist.php

Displays a list of locations. This should be divided into area, street and location, to provide an easy way to find the right location.

editlocation.php

Display and edit a location, including information about street and area.

Parameter *id* - Location id. If not defined, create a new location.

locationmap.php

Generates a map of locations. This could be a link to a map site such as Eniro or Hitta.se.

Parameter *locid* - Location id of a location that should be highlighted.

E.7 Administration**administration.php**

Startpage for the administration

contractlist.php

Displays a list of contracts.

Parameter *customerid* - Customer id of a customer for which contracts should be listed.

contract.php

Display and edit a contract. It also provides an overview of included services.

Parameter *id* - Contract id. If not defined, create a new contract.

Parameter *customer* - Customer id to bind a new contract to.

srv[subnet|email|domain|tv|internet|connection|other].php

One page for each service type. Display and edit the service.

Parameter *id* - Service id. If not defined, create a new service.

Parameter *contract* - Contract id to bind a new service to.

E.8 Technical**technical.php**

Startpage for the technical subsystem.

nodes.php

Displays a list of nodes. This page should include a search function to find nodes. It should display some quick status information for the nodes and the nodes location.

node.php

Display and edit a node. This page may require dynamic content depending of nodetype. It should also provide an overview of node services and statistics collection.

Parameter *id* - Node id. If not defined, create a new node.

Parameter *nodetype* - Nodetype for the new node. Optional.

nodeservice.php

Display and edit a node service.

- Parameter *id* - Nodeservice id. If not defined, create a new node-service.
- Parameter *nodid* - Node id to bind the new nodeservice to.

statistics.php

Edit and display a list of statistics settings for a node.

- Parameter *id* - Statistics id. Identifies a specific statistics entry. If not defined, create a new statistics entry.
- Parameter *nodid* - Node id to bind a new statistics entry to. This could also be used to filter the list on a specific node.

nodetypes.php

Displays a list of nodetypes.

nodetype.php

Display and edit a nodetype. It should also provide a list of the node ports and SNMP MIBs, which should be editable.

- Parameter *id* - Nodetype id. If not defined, create a new nodetype.

links.php

Displays a list of links. The list should be possible to filter on startnode and endnode.

link.php

Display and edit a link.

- Parameter *id* - Link id. If not defined, create a new link.

vlans.php

Displays a list of VLANs.

vlan.php

Display and edit a VLAN. A vlan consists of a set of links, so this should provide a list of selected links, which should be editable.

- Parameter *vlanid* - VLAN id. If not defined, create a new VLAN.

connections.php

Displays a list of connections. The list should be possible to filter on a included node or link

connection.php

Display and edit a connection. A connection consists of a set of VLANs and/or a set of links.

Parameter *connid* - Connection id. If not defined, create a new connection.

linkmap.php

Displays a topology map of all links in the system. This should be automatically generated from the database. It should be clickable and it should be possible to highlight and select a connection, a VLAN or a link.

Parameter *connid* - Connection id of a highlighted connection.
 Parameter *vlanid* - VLAN id of a highlighted vlan.
 Parameter *linkid* - Link id of a highlighted link.

E.9 Tasks

tasklist.php

Displays a list of current matters, including the last event of each matter.

Parameter *department* - Department id used to filter the list on the next department to handle the matter.

task.php

Displays a matter, including all events. Possibility to add a new event is necessary.

Parameter *id* - Event id. If not defined, create a new event.
 Parameter *linkinfo* - If defined and a new event is created, add this to the link field.
 Parameter *department* - Filter displayed events to this department id only.

pm.php

Displays personal messages. Provides a fast and easy way to create a new pm or reply to an old.

Parameter *op* - Operator id. If defined, create a new pm with this operator as receiver.

E.10 Statistics and surveillance

getnodestatgraph.php

Generates a graph from collected statistics. This uses RTGplot to generate the figure, By using this method, the graphs can be referenced to by node id and a few parameters.

Parameter *nodeid* - Nodeid to generate a statistics graph for.
 Parameter *mibtype* - MIB type to generate a graph for.
 Parameter *params* - One or more parameters necessary for RTGplot.

surveillances.php

Lists available surveillance script instances.

surveillance.php

Display and edit a surveillance script instance.

Parameter *watchid* - Surveillance id. If not defined, create a new surveillance script instance.

survscripts.php

Displays a list of available surveillance scripts.

survscript.php

Display and edit a script. This should provide test and save functions to be used during development of a new script.

Parameter *scriptid* - Script id. If not defined, create a new script.

cronexecute.php

Executes surveillance scripts according to defined runtimes. The information is collected from the database.

Parameter *watchid* - Surveillance id for the script instance to run.

class SaS

This class contains functions to generate run scripts for the two systems.

sas.php SaS.updatestatbatch

Updates the run script file for the statistics tool RTG.

sas.php SaS.updatecrontab

Updates the crontab file for the surveillance system.