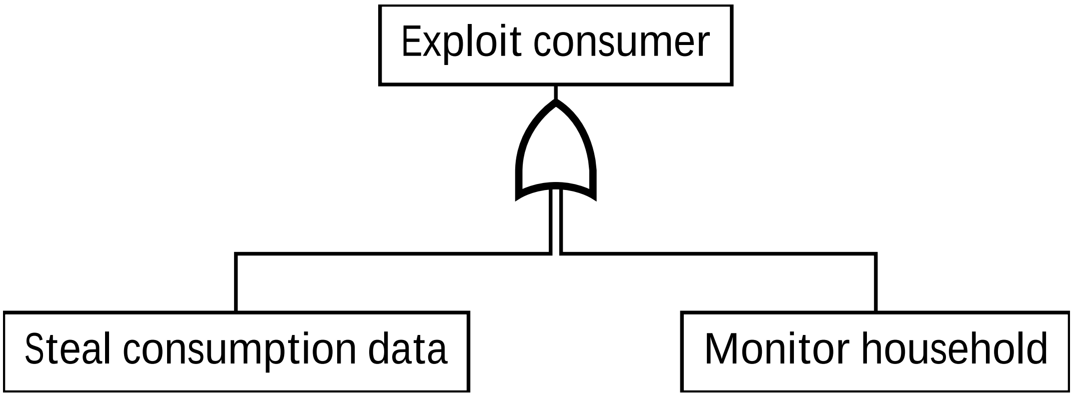
Smart Meter Security Analysis



Project Report

DES904E15

Aalborg University
Department of Computer Science

AALBORG UNIVERSITY
STUDENT REPORT

Department of Computer Science
Aalborg University
http://cs.aau.dk

Title:
Smart Meter Security Analysis

Theme:
Distributed and Embedded Systems Prespecialization

Project Period:
Fall Semester 2015

Project Group:
DES904E15

Participant(s):
Bruno Thalmann
Mikael Elkiær Christensen
Mikkel Sandø Larsen
Stefan Marstrand Getreuer Micheelsen

Supervisor(s):
René Rydhof Hansen
Mads Chr. Olesen

Copies: 7

Page Numbers: 87

Date of Completion:
January 14, 2016

Abstract:

Smart meters will be implemented in EU by 2022, and has to make the electrical grid more flexible. But the speedy implementation may cause the security aspect to be less prioritized than what is desirable.
This report starts out by examining the smart meter and smart grid as well as the problems already known to exist in smart meters. Because of the loosely defined laws in the area, as well as the ongoing implementation, the report defines a smart meter context model that will be used as reference throughout the rest of the report.
Thereafter the report will investigate available security models in order to determine if one of the already existing models can be used to model the system. The applicability of each model will be discussed in order to aid future work on the subject. Afterwards the smart meter context will be analyzed and possible attacks on the smart meter infrastructure will be presented in attack trees. Attacks will be detailed and discussed and finally we will present possible paths forward.

Contents

Preface

This report has been prepared by 9th semester Software Engineering students at Aalborg University, during the fall-semester of 2015. It is a pre-specialization project, which should serve as a the preliminary work for one or more master theses. It is expected of the reader to have a background in IT/software, due to the technical content.

References and citations are done by the use of numeric notation, e.g. [1], which refers to the first item in the bibliography.

We would like to thank our supervisor René Rydhof Hansen and co-supervisor Mads Chr. Olesen for their excellent supervision throughout the project period.

Aalborg University, January 14, 2016

Bruno Thalmann
<bthalm11@student.aau.dk>

Mikael Elkiær Christensen
<michri11@student.aau.dk>

Mikkel Sandø Larsen
<milars11@student.aau.dk>

Stefan Marstrand Getreuer Micheelsen
<smiche11@student.aau.dk>

Introduction

Energy consumption is ever increasing and our current electrical grid is outdated, as it is unable to handle the more dynamic production and consumption of power we have today. There is also an increasing generation of renewable energy (e.g. wind and solar), but optimal use of this is limited by our current electrical grid. As it is now, renewable sources are often shut down if they are producing more than can be consumed. On the other hand, during periods with little renewable power, more expensive and environmentally unsafe sources are used. Additionally, most suppliers only support very simple tariffs with one fixed price, or a single division of prices, throughout the day.

The implications of the above statements have led to an increased focus on energy usage, with optimizations of the electrical grid as a high priority. This is why, by 2022 (80% by 2020), all electrical meters in EU member countries must be replaced by *smart meters* [3, 29]. An expansion of the electrical grid, as it is extended to connect more EU member countries, along with the addition of smart meters – enabling home owners to supply themselves and others, will form a *smart grid*.

The suddenness and size of this project has the potential to bring a lot of security and privacy-related issues with it. The millions of smart meters placed in private homes throughout Europe can lead to mass privacy breaches, as smart meter measurement data mining can reveal much about each individual household. Parties with malicious intent can use any ill-gained access to smart meters or other smart grid components to possibly shut down or in other ways control the electrical grid.

This report will have a general focus on these issues and investigate the security of the smart meter system. Firstly, in Chapter 1, the context in which we operate, and possible issues, will be elaborated, concluded by some assumptions about what kind of system will be implemented – with a base in the Danish solution. Then in Chapter 2 several security models will be presented, which could have a potential interest in ensuring that smart meter solutions are secure. To get a better idea about what kind of attacks are possible in a smart meter system, Chapter 3 explores possible attackers and forms attack trees – which are used to further identify concrete attack techniques. These attack techniques are then elaborated in Chapter 4, with a technical presentation based on current security engineering. Finally, we conclude in Chapter 5 on the achievements of this report, and what work from now on can be done to further improve the security in the coming smart meter systems.

Chapter 1

Context

In order for us to discuss potential problems with any smart meter system, we first need some background knowledge. In this chapter we will first of all very shortly present what the smart grid and smart meters are, along with a short outline of potential problems mentioned in the litterature. Two big issues, privacy and remote access-ability, will be further elaborated as these are two of the biggest issues with any smart meter system. Finally, some assumptions will have to be made about any future implementations in order to more concretely discuss any security-related problems and/or solutions, which is done by presenting a “Smart meter context model”.

1.1 Smart Grid and Smart Meters

A smart grid is an electrical grid supported by a net, allowing two-way communication, whereas earlier it was only one-way. This allows for much more dynamic power supply and consumption, as suppliers will know more about their consumers, and consumers will have more options in regards to their consumption.

The outline of a smart grid can be seen in Figure 1.1. This consists of three main actors:

- Power suppliers – windmills, solar panels, traditional power plants and external suppliers.
- A net supplier – *Data Hub*[1] and smart meters.
- End consumers – smart homes/buildings.

[1] Danish term from [32]

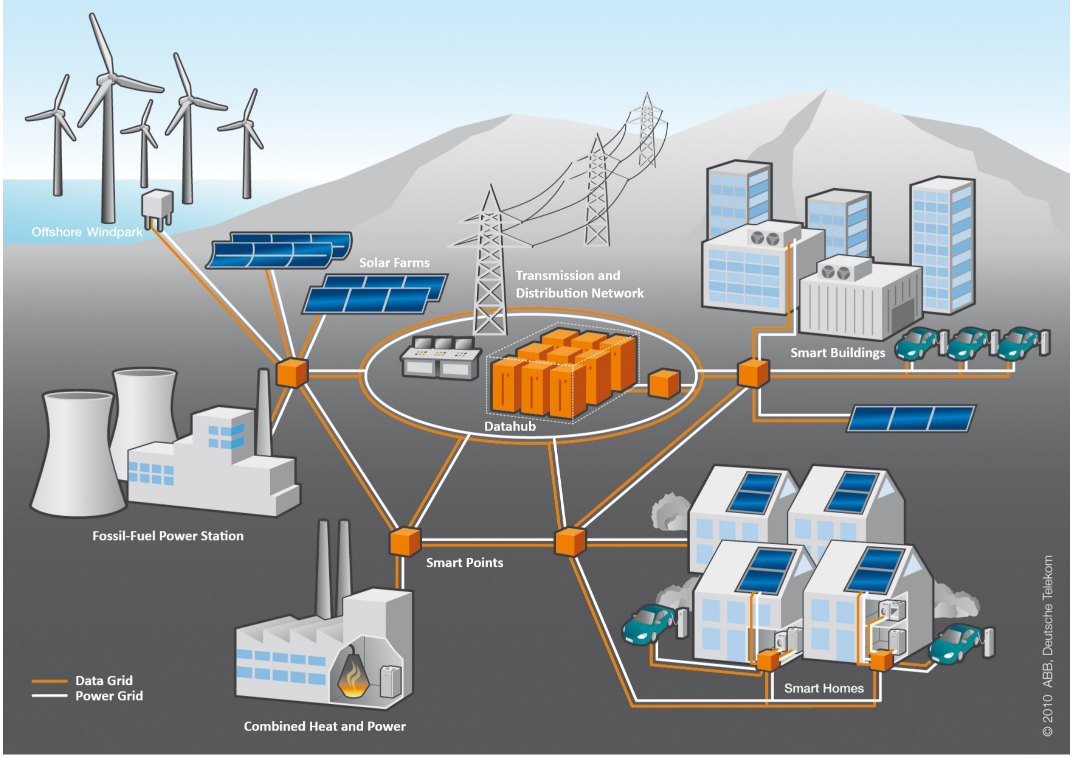


Figure 1.1: Smart grid outline[2]

A smart grid has several advantages over the old grids [16, 29]. For instance, it will enable prices to be base on supply and demand as well as the power source available. Consumers will also be able to connect their own power generators, such as windmills and solar panels. Prices will then more realistically reflect environmental strain, which can make the consumer more likely to consider the environment when consuming power.

The use of smart meters will also provide more detailed information about the power consumption of the consumer, which enables the consumer to use smart appliances that can turn on when the power is cheap. The consumer can also monitor the power usage and adjust his practices according to the prices.

On a large scale, the smart grid also enables connecting the national grids across Europe for better utilization of the generated power.

1.1.1 Smart appliances

Smart appliances[15] are ordinary appliances, such as air conditioners or dishwashers. What makes them smart is the fact that the times or periods in which they are turned on are highly configurable, so that they can match changing power prices. This can be done manually, by the user looking at current pricing and setting a timer when to

[2] https://www.telekom.com/medien/bild-ton-und-infografiken/infografiken/155030

turn on. However, some smart appliances can also by themselves look up prices and choose appropriate times to turn on.

1.1.2 Problems

In regards to enabling an EU-wide smart grid, there are bound to be problems, as this is an immense project. The roll-out will not be the same in every nation, as each nation has its own infrastructure and legal constraints. Additionally, they differ in which parts of their electrical grids are privately and publicly owned. However, some shared problems still exist, such as privacy, conflicts of interest, and attack vulnerabilities [1, 3, 2].

Different architectures

Even though the implementation of smart grid and smart meters is an EU project, they are not that specific when it comes to where responsibilities should lie. It is up to each nation, depending on their current system and infrastructure, who should distribute smart meters, and who will own the data.

Here are a few examples of different architectures [3]:

- **Italy** – Regulated monopoly (supplier and distributor is the same).
- **Germany** – Free market for both distributor and supplier, households will have free choice in both.
- **UK** – Centralized government-licensed monopoly, which will have a data hub from which data will be distributed to suppliers, customers, etc.
- **Denmark** – Centralized through 60 government-regulated distributors, a government-owned data hub contains all data [14, 32].

Privacy

With the adaptation of smart meters, it will be possible to collect power usage data more often. This is possible as it can be done remotely, whereas earlier a display would have to be read on the physical electrical meter. It also makes sense to have more readings, as tariffs will vary more [29, 12], and therefore more measurements are needed to match prices during tariff-determined periods.

This possibility of finer granularity meter readings can be exploited in several ways and by different actors. The supplier would like this information in order to better tailor prices for a certain customer, thus limiting competition as their competitors do not possess this information. If the power usage data was obtained by an electronics vendor, they could use it to target certain advertisements, based on what devices they detect (or do not detect) through the power usage. Anyone with malicious intent could use the power data to determine who, if anyone, is home (e.g. burglars). Finally, the government could use this data to surveil the public.

Some important issues with smart meters and their measurements are therefore:

- Who owns the data?
- Who should be able to access the data?
- With which granularity should supplier/government/user be able to access the data?
- How do we ensure that only the correct people have access to the smart meter and its data?

Conflict of interest

The various actors involved with smart grids have different interests and potential gains. Governments generally want lower consumption for environmental reasons. They especially want consumption down during peak hours, in order to avoid the usage of environmentally unfriendly energy sources. Depending on tariffs, suppliers might also want power consumption down during peak hours if they cannot provide enough, from their perspective, cheap energy. However, suppliers generally want high consumption, so that they can sell more power and thus make more money. Consumers want to use power as needed, but at lower costs.

These interests exemplify considerations by actors in a smart grid system. Naturally there could be more such as consumers and suppliers having environmental considerations or governments seeking to increase productivity on a national level with lower regards for the environment.

The above relations spark several conflicts of interest:

- Should governments be able to control the power consumption of consumers? If so, how?[3]
- Should others than the user be able to turn off the power? (See Section 1.2.2) If so,
  - who should be able to use this?
  - how should it work? (With abuse in mind.)

Vulnerabilities

In the current system consumers are already tampering with their mechanical electrical meters. Turning these mechanical meters into smart meters will possibly remove some attacks, but it will definitely open up for new attacks. Switching to smart meters introduces a new branch of attacks; digital attacks. These include attacks that are general to any publicly exposed unit, or any unit that sends data over public networks.

[3] E.g. carbon rationing in the UK [2]

1.2 Issues Mentioned in Litterature

During our search through litterature, two problems stood out – privacy and remote access-ability. In order to examine the severity of these issues we have found sources that investigate the topics further. These will be presented in the following.

1.2.1 Smart meter privacy concerns

The consumption data can be collected in various granularities. Sensitive knowledge about the consumer can be obtained from this data if it is fine-grained enough. The following describes how one can obtain and use this knowledge, which is based on Molina-Markham et al. [25].

Experiment

The setting, in which sixty days of power consumption data from three households was collected, consists of the following physical components:

- TED energy monitor
- SheevaPlug

The energy monitor is connected to the two incoming electricity phases, A and B.[9] The SheevaPlug is connected to the router and the energy monitor, and is used to access the data remotely. The energy monitor returns a tuple (t, p), where t is the time and p is the power-usage since last measurement. The power is measured every second, so we have a tuple for every second with the power used this past second. Finally, the three households are to make a "power activity"-journal for a minimum of three days. In this journal, the individuals of the household log whenever they turn on and off any electric devices.

When the data is collected it is analyzed in four steps:

1. Pre-process data
2. Tag power events
3. Filter out automated appliances
4. Map consumption events to real life events

Pre-process data The data is pre-processed using DBSCAN, which is a density-based clustering algorithm. This helps group power tuples into power segments, where a power segment is a collection of power tuples with a pattern adjacent in time. Each power segment then gets tagged with a label, start time, average power usage, duration, beginning power step, and a shape label.

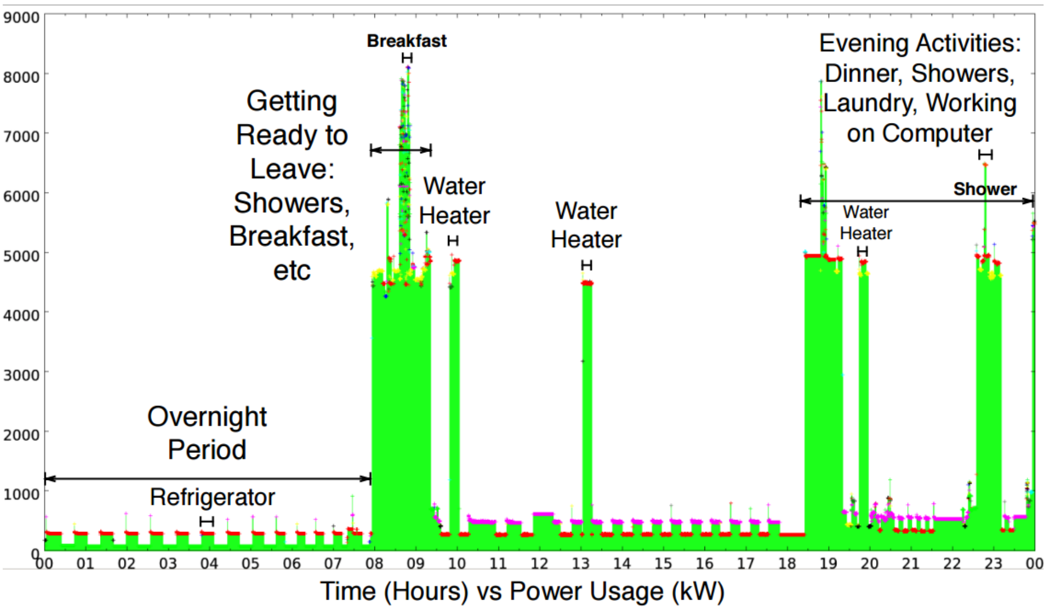


Figure 1.2: A day of consumption data after step 2 of analysis, labelled with real life events – taken from the power activity journal

Tag power events In Figure 1.2 we are able to see the consumption data for one household after step 2 of the analysis. On the x-axis we have time in hours, for an entire 24-hour period, and on the y-axis we have power usage in kWh. Already at this stage of the analysis, we are able to say something about when there is activity in the household. We can also see how there are some automated appliances that are on all the time or at certain intervals. We can determine this by assuming there is almost no activity in the night, and by collecting data over several days.

Filter out automated appliances Now automated appliances, such as the refrigerator (see Figure 1.2), can be filtered out. This is done by looking at the consumption data and reason about whether the power usage is from a human interaction or not. If, for instance, the distinct usage also occurs at night, it is probably an automated appliance and by looking at a lot of data one can be rather certain.

Map consumption events to real life events In Figure 1.3 we can see what the consumption for a small time period looks like after all four steps of the analysis. At this point it is pretty easy to say something about what kind of activities are going on. The time period is for a typical morning, where we can see the household uses the stove, coffee maker, toaster, and some computer screens.

As we can see from this example, when collecting only a small amount of consumption data and doing trivial analysis, it is possible to say a lot about a household.

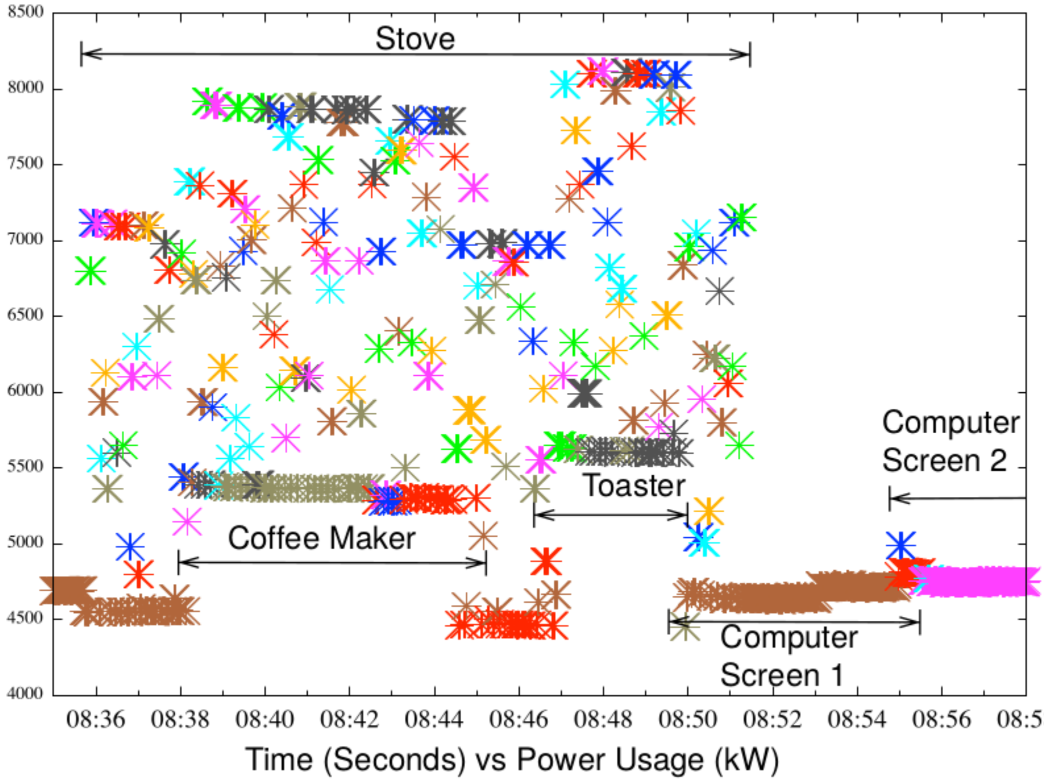


Figure 1.3: Power consumption data after all four steps of the analysis for a small time period – a morning when the household is getting breakfeast, etc.

Utilities companies or similar, that have access to much more data, can learn even more by using their combined knowledge.

Privacy issues

As it can be seen in the previous section, it is possible to determine a lot about a household, simply based on power consumption. This can lead to privacy issues. Some of these issues will be described in the following.

Amount of people home When having a lot of consumption data one can determine the amount of people who are home. One can thereafter begin to identify patterns in the power usage and map that to individual people from the household. This means that over time it is possible to determine *who* is home.

Household activities It is also possible to say a lot about household activities, even with a small amount of consumption data. For instance, we can begin to reason about if a household had a good night's sleep, or if the household was eating hot or cold breakfast, and even if the household was watching a specific sport event last night.

Information about appliances By extending the analysis, one could look for specific appliance signatures in the power trace, and from this say even more about the household.

This approach has been done by Parson et al. [30]. They use prior knowledge about appliances in the household to find the signatures of appliances in the power trace. They then use these power signatures to dis-aggregate the power trace into appliances. They also tested this approach as a live example in the real world, at six households in the United Kingdom.

1.2.2 Concerns regarding the remote access-ability of smart meters

As the smart grid and smart meters are getting more common, an important observation was made by Anderson and Fuloria [1]. This observation is regarding the possibility of the electrical company to switch off the smart meter. The purpose of a remote off-switch could be to switch of the electricity if a customer is not paying. When having smart meters controlled one place, this place could be a target by terrorists, other countries, or by a group of activists.

Large scale consequences In a scenario where millions of smart meters are deployed, and controlled from a central control unit, not taking possible attacks against the off-switch into account could be very dangerous. Anderson and Fuloria [1] state that this is happening in the United Kingdom. This means that there probably are a lot of vulnerabilities in the setup. If an attacker gets access to the central control unit, he has the ability to compromise all the connected smart meters. By also tampering the cryptography of the smart meters, the attack could last for weeks for some consumers. People would die from hypothermia because of lack of electricity at hospitals. The region it would effect would be in chaos.

Attackers Possible attackers for such an attack could be:

- States – when there is international tension.
- Terrorist organisations.
- Environmental activists who are frustrated with governments not taking the environment seriously.
- Criminals who switch off all or some of the smart meters from an electrical company and demands money for switching them on again.

1.3 Smart Meter Context Model

As no fully implemented smart meter systems currently exist, a lot of assumptions will have to be made before it is possible to imagine how such a system could be attacked. We do this in two steps.

The first step taken is to outline the system as closely as possible. Since there is no single correct system architecture, we have chosen the one opted by our native country: Denmark. This system outline is based on the sources and information gained in the previous sections.

1.3.1 The smart meter system

The system is pictured in Figure 1.4. The black lines indicate information flow between actors in the system. The *consumer* can interact, through a client, with the *smart meter* in order to control *smart appliances* and monitor the production of his *home production* devices. The *data hub* stores the consumption data of the consumer. This data is available for the *consumer*, the *distribution companies* and the *electrical companies*. These three actors have permission to see the consumption data in different granularities. The consumer can see the data in the finest granularity, whereas the distribution companies and the electrical companies has access to a coarser granularity that makes it possible for them to calculate the bill. The electrical companies send price information to the data hub, which the consumer can see through the smart meter.

Power in the system is indicated by a red line. Power flows from a power source and from home production to the smart meter which powers appliances in the home.

Finally, the *smart meter manufacturers* have physical access in order to install and maintain smart meters, as well as the ability to update firmware. This is indicated by the blue line.

1.3.2 Actors and threats

Placing a smart meter in a consumers home and simultaneously requiring external access to its data is a complicated task with many potential risks. Figure 1.5 represents this context and the actors therein. This has been used as a starting point for a brainstorm that maps the potential attacks on the system.

Actors Below is the list of the actors represented in the full system and context. The list below will briefly describe the actors, their objective, and how they can abuse a smart meter to achieve their objective.

- **Consumer**
  The resident of the depicted home and the one whose power consumption is monitored. The smart meter is installed in the consumer's home. The consumer may want to attack his own smart meter in order to reduce his power bill.

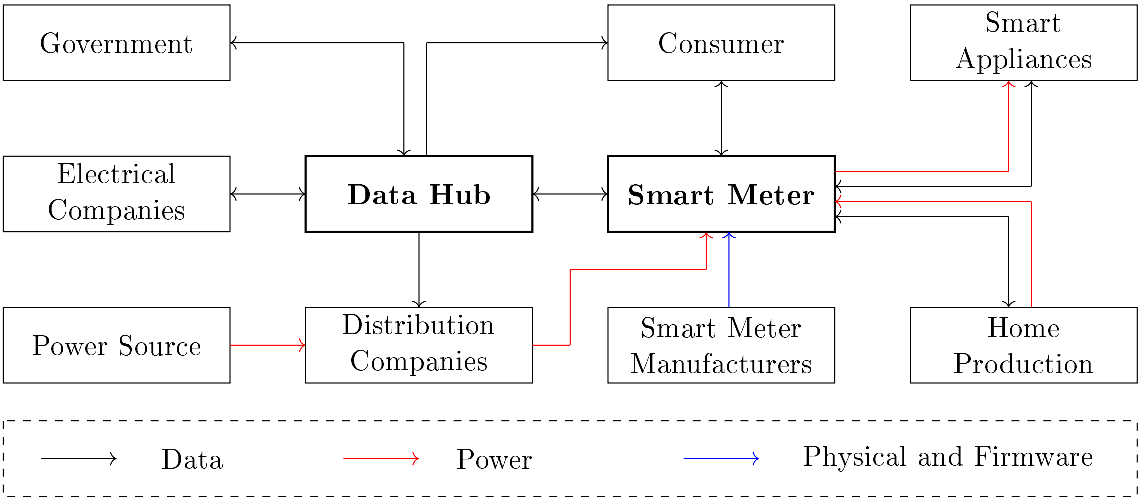


Figure 1.4: The smart meter system[31]

- **Burglar**
  A burglar can use the smart meter to find out when the consumer is home, in order to break in, as well as finding out which products the consumer owns, in order to assess him as a target.

- **Neighbor**
  The next door neighbor to the consumer. We assume that the neighbor will perform attacks on the consumer in order to remove annoyances, such as loud appliances or music. In case of neighborly disputes, messing with the SM and connected appliances could also be viable strikes.

- **Partner**
  The consumer's partner. In this context there is no distinction between a partner who is living with the consumer and one that is not. The partner would like to surveil or get revenge over the consumer, in case of misdoings on the consumer's part – such as adultery.

- **Electrical Company**
  The company billing the consumer for his power consumption. The electrical company may want to increase the bill of the consumer.

- **Distribution Company**
  Manages the part of the grid closest to the consumer and provides him with electricity. Installs the smart meters in the consumer's home. Government-controlled intermediary between the electrical company and the consumer. The distribution company therefore has no gain from abusing the smart meter.

- **External**

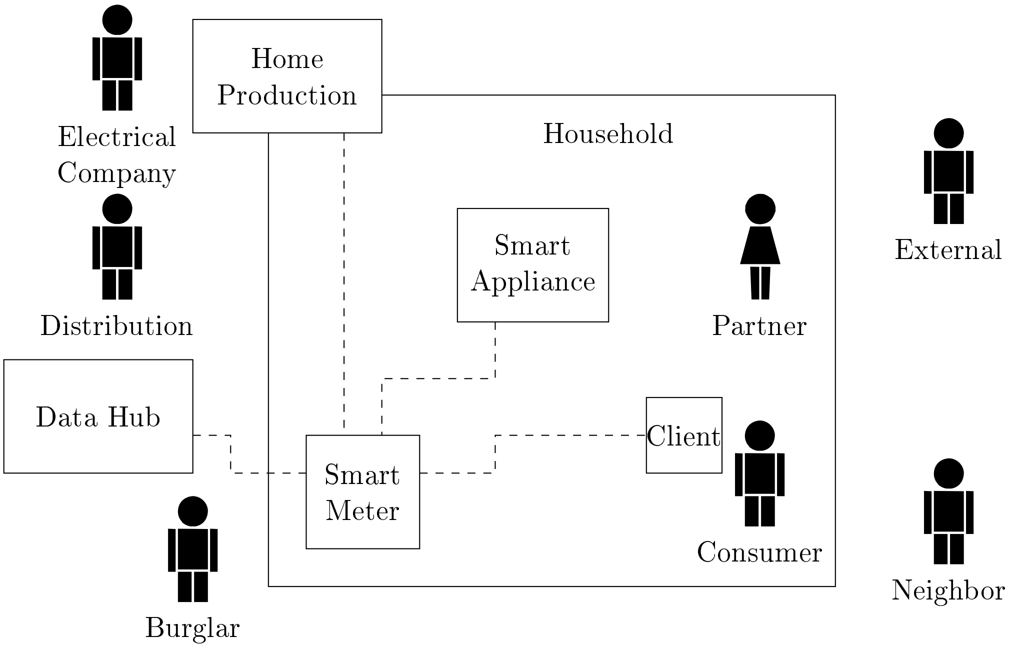


Figure 1.5: The smart meter context model

- **Information gatherers**
  Some companies depend on information about consumers and can use the information provided by the smart meter to profile consumers.
- **Developers of third party apps**
  The developer of apps that the consumer will use for monitoring his electrical consumption. These apps can be either mobile, desktop, and web applications. The information gatherer can abuse this position in order to collect data about the consumer and possibly sell it.
- **Appliance company**
  The provider of home appliances that can connect with the smart meter. Can collect data through appliances and possibly sell it.
- **Government**
  The government officials (including counties and municipalities).
- **States**
  In countries with tension between them, one country could have an interest in abusing smart meters to attack citizens of the opposing country.
- **Terrorist organisations**
  Terrorists can abuse the smart meter for terrorist acts, like shutting off power in a city or region.
- **Environmental activists**
  Activists can abuse the smart meter as a means of demonstrating environ-

mental politics.

  - **Blackmailer**
    A criminal who switches off all, or some, smart meters from an electrical company and demands money for switching them on again.

Objects

- **Smart Meter**
  Controls and records power consumption for home appliances.
- **Smart Appliance**
  Optionally controlled from the smart meter. For instance a washing machine can be started and stopped when scheduled.
- **Data Hub**
  Consumption data is stored on the Data Hub, which makes it available for the distributor and the electrical company.
- **Home Production**
  The consumer can connect his own home production devices such as solar cells or windmills.

Smart Meter

The following describes the conceptual smart meter we have used in our considerations, based on Section 1.1, Pedersen and Sørensen [31], and Anderson and Fuloria [3]. These assumptions will be used throughout the report.

- The smart meter is installed in the household of the consumer, in this case a house.
- Home appliances are connected to the smart meter and can be managed from it.
- The smart meter can be accessed through an API.
- The consumer can access and manage his home appliances connected to the smart meter through some sort of program, e.g. an app developed by a third party.
- The smart meter manufacturer has the ability to update firmware.
- The home appliance company can update the firmware of their appliances through the smart meter.
- The smart meter sends its consumption data to the Data Hub.
- The electrical company have the ability to switch off the smart meter

- The electical company obtains billing information from the Data Hub.
- Home production of electricity, for instance a windmill, is also connected to the smart meter.

Chapter 2

Security Models

As was seen in the previous chapter, the implementation of smart meters can lead to problems related to both security and privacy. There are many stakeholders who should, and others who should not, in some way or another be able to access the individual consumer's smart meter. Therefore we explore the possibility of using established security models to ensure security or privacy is not violated. The security models will be presented in chronological order of publication, in order to show the development of the models through time.

Firstly, we will present the Bell-LaPadula Security Policy, which is a military-developed model mainly concerned with controlling information flow and authorization. Then we will present a more general approach of describing protection systems, so that it can be shown whether the described system is secure or not. As an alternative to Bell-LaPadula we will present The Chinese Wall Security Policy, which is a commercial security model concerned with more dynamic authorization. Finally, we present a more modern security model with the Decentralized Label Model with main concerns similar to Bell-LaPadula.

2.1 The Bell-LaPadula Security Policy

In 1973 Bell and LaPadula devised a mathematical model intended for use in military and governmental computer systems. The model formalizes users accessing data and how to handle this in a secure way, so confidential information cannot be leaked to a lower classification level. The following description is based on LaPadula and Bell [22].

2.1.1 Security

Before we go into details about how the Bell-LaPadula (BLP) security model is applied to computer systems, we will first discuss some important concepts.

In any system we will have a set of objects O and principals S. Objects are entities that can somehow be manipulated by principals. Principals are processes or programs in execution that represent a user or a group of users.

Classification and Clearance Attached to any object or principal will be a classification or clearance respectively. We denote the set of all classifications/clearances $C = \{C_1, C_2, \ldots, C_q\}$ where $\{C_1 > C_2 > \cdots > C_q\}$ holds, giving a hierarchical sequence of classifications/clearances. This means that for any object $o \in O$ with *classification* $C_i \in C$, where C_i is arbitrary, any principal $s \in S$ will need a *clearance* $C_j \geq C_i$ in order to access o.

Category and Need-to-know In addition to classification, objects can belong to one or more categories, which can be seen as security groupings of certain objects. Similarly, principals can have one or more need-to-know, which are the security groupings for principals. The security categories are a set $K = \{K_1, K_2, \ldots, K_r\}$ and unlike classifications/clearances there is no hierarchy of categories. In order for principal s to access an object o with *category* K_m and K_n, where K_m and K_n are arbitrary, s must have both *Need-to-know* categories K_m and K_n.

Classification and category vectors In the model, classifications and categories are stored in vectors. Four functions are used for lookup in these vectors. f_1 performs a lookup in the principal clearance vector, f_2 in the object classification vector, f_3 in the principal need-to-know vector, and finally f_4 performs a lookup in the object category function.

Visualization This kind of security system can be seen as a *lattice*, see Figure 2.1. Each node represents a *security level*, which we define as a combination of classification/clearance and a set of categories.

A principal with the clearance *Top Secret* is able to access any object with the security level *Top Secret* and all those below (*Secret* and *Unclassified*). On the contrary, a principal with clearance *Unclassified* can only access objects with the security level *Unclassified*.

Similarly, the category(ies) of each node limit what objects a principal can access. If we have a principal s with clearance *Top Secret* and category *Crypto*, s will have access to any objects with security level $(TopSecret, \{Crypto\})$ and those below.

2.1.2 Access attributes

The model considers four attributes for access in a complex computer system: *read-only*, *read/write*, *execute* and *append*. In addition *control access* is used to give attributes to other users. Formally, the access attributes are defined as a set in which each member corresponds to the aforementioned attributes:

$$A = \{r, w, e, a, c\}$$

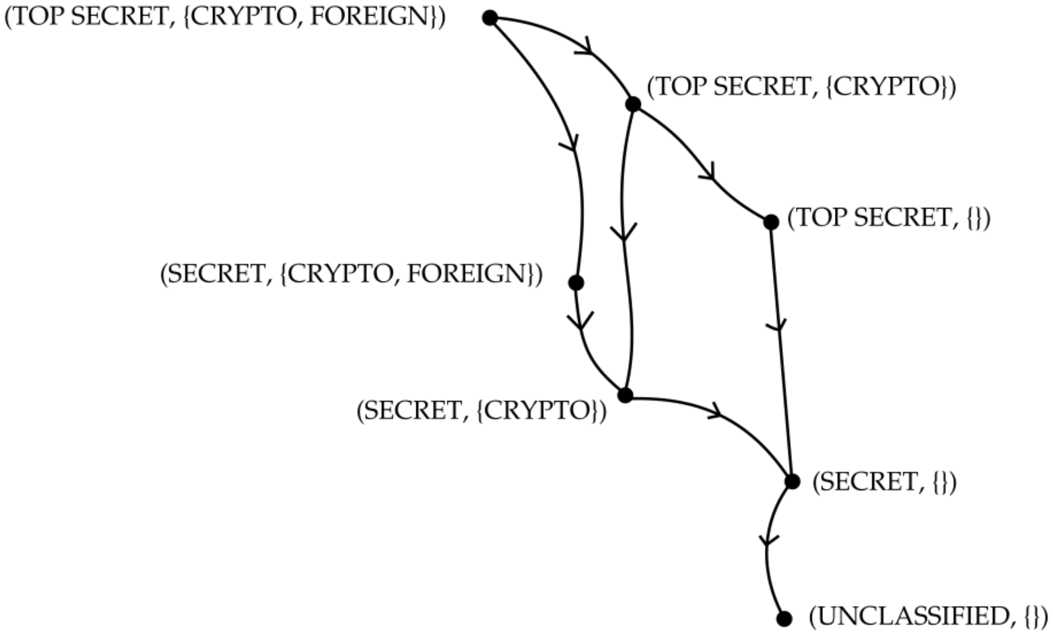


Figure 2.1: A BLP lattice [4]

Read-only This attribute makes it possible to read the object but not alter it. The classical example is a file that contains information that should not be changed. Alternatively it could be a list containing the principals in the system with their clearance levels. A principal of low clearance should be able to read this list, but not change it.

Append Append describes a pure write operation. This means that it is possible to append information to the end of a file without being able to extract information about the rest of the file.

This can also be used with a printer which appends information about what is being printed. By doing this it is sufficient that the classification of a piece of information is matching the classification of the printer in order to prevent unauthorized personnel from reading the information.

Execute The execution attribute makes it possible to execute an executable file. If the principal does not have permission to read from or write to the file he will only be able to execute it. Similarly, the executable file can produce output that is of a higher classification level than the clearance level of the principal executing it.

Read/write This attribute signifies that read and write access are both allowed. This attribute is what is traditionally used when editing text files.

Control access The control access attribute models the notion of a principal having control over a file. Having this attribute a principal can give the four attributes above to other principals in the system.

2.1.3 Requests and decisions

In a computer system the principals are not directly accessing objects in the system, it is processes in the system that act on behalf of the principal. In the following, a user *requesting* access to a file will be written as a principal requesting access and the response to this request a *decision*.

> **Definition 2.1 (Access matrix)**
> Let S and O be the principals and objects (respectively) in a system. Then M is an $|O| \times |S|$ matrix representing the current access attributes, and entries $M_{i,j} \subseteq A$ (see Section 2.1.2) represent the access attribute that S_i currently has for O_j.

Requests There are 4 possible requests that can be made by any principal:

1. be granted access to an object in a particular mode.
2. another principal should be granted access to an object in a particular mode.
3. create an object in the system.
4. delete an object in the system.

In order to carry out the second request, the principal needs control access of the object that is being requested access for, as well as the particular mode. So that for a principal S_i requesting read access r for principal S_k for object O_j, S_i must have both r, as well as control access c, for O_j itself. Control access is only given to a certain object when the object is created. Unlike the other attributes, control access cannot be given by any principal to another principal.

When a new object is created, it is initially inactive. In addition to the 4 requests above, in regards to creating objects, we also have the following 2 additional requests:

1. alter classification and category assignment of an unused object.
2. activate an unused object.

For the 4th request, deletion, the object deemed for deletion is inactivated and all current and future privileges are revoked.

Decisions Any request made will be met with a response of values *yes*, *no*, ?, or *error*. *yes* and *no* responses are given when the request is recognized and a check can be performed, thus giving a response of *yes* if granted and *no* if not. *error* is given in case the request is recognized, but there cannot be given a proper response due to more than a single rule can be applied. ? is given in cases where the request was not recognized.

2.1.4 Preventing security compromise

In order to ensure that data cannot be compromised, the previous definitions of access attributes and requests can be utilized to formalize properties that ensure that compromise cannot occur.

Security condition

The security condition states that a principal with a given clearance level is prevented from having read access to any object, which is, or can be, a source of information with a classification level that is higher than the clearance level of the principal.

Formally, this can be expressed as the following:

Definition 2.2 (Security condition – entry)
Let $M_{i,j}$ represent an arbitrary entry in an access matrix, as defined by Definition 2.1. Then, if and only if

- $M_{i,j} \cap \{execute, append, control\} \neq \emptyset$, or
- $M_{i,j} \cap \{read, write\} \neq \emptyset \wedge f_1(s) \geq f_2(o) \wedge f_3(s) \supseteq f_4(o)$

the entry satisfies the security condition.

Definition 2.3 (Security condition – system)
An access matrix M satisfies the security condition if

$$\forall S_i \in S : \forall O_j \in O : M_{i,j} \text{ satisfies the security condition}$$

We say that a system is secure if its access matrix satisfies the security condition for all its states.

Seen in context of the lattice in Figure 2.1, the security condition can be seen as the disallowance of any principal s from reading any objects above its security level. This is often mentioned in other literature as *no read-up*.

*-property

To simplify the description of the *-property we wish to examine which objects a principal has a specific access to. That is, given a set of attributes $a_1, a_2, \cdots, a_n$, we are interested in the objects that a given principal has either of these access attributes to. This can be expressed formally as:

> **Definition 2.4 (Matrix extraction)**
> Let S represent a principal and $x, y, z \in A$ a set of attributes. Then
>
> $$b(S : x, y, \cdots, z) = \{o : o \in O \wedge [x \in M_{S,o} \vee y \in M_{S,o} \vee \cdots \vee z \in M_{S,o}]\}$$
>
> represents all the objects to which S has any of the $x, y, \cdots, z$ access attributes.

The *-property ensures that principals can only write information to an object o_1 if that information is read from an object o_2 which is at the same or a lower *security level*. This property must apply to all objects that the principal has access to.

The following definition expresses this formally:

> **Definition 2.5 (*-property)**
> Let S be a principal. Then $O_{write} = b(S : w, a)$ is the set of objects to which S can write information and $O_{read} = b(S : r, w)$ the set from which it can read. Using these sets we can define the *-property for S as:
>
> $$\begin{array}{l} \mathbf{T} \quad \text{if } O_{write} = \emptyset \vee O_{read} = \emptyset \text{ , otherwise} \\ \forall o_w \in O_{write} : \forall o_r \in O_{read} : f_2(o_w) \geq f_2(o_r) \wedge f_4(o_w) \supseteq f_4(o_r) \end{array}$$
>
> - A state of an access matrix satisfies the *-property if an only if for each $s \in S$ the following proposition holds.
> - We say that a system satisfies the *-property if its access matrix satisfies the *-property condition for all its states.

When seeing the *-property in the context of Figure 2.1, the *-property states that it is not allowed for any principal s to read any object o_1 and write any of this information to an object o_2, if o_2 is not above, or at the same security level as, o_1. This is often mentioned as *no write-down*.

2.2 Protection in Operating Systems

As most protection systems have different approaches to how they manage rights and chains of trust they are hard to compare on equal terms. Additionally, it is often not easy to express which rights are leaked under certain conditions, or to whom they are leaked. Oftentimes such leaks are only described informally by examining key features of a given system.

The model described in Harrison, Ruzzo, and Ullman [20] from 1976 can be used to solve these problems. By formally describing protection systems using a unified model, they can be compared in terms of the rights that they leak. Using this approach two possible solutions (protection systems) to a problem can be compared on even terms.

In the following the model proposed by Harrison, Ruzzo, and Ullman [20] is presented.

2.2.1 A protection system

A protection system can be described in terms of a set of rights R and a set of commands C. Both these sets are finite and static for a given system. Because of this, a system cannot dynamically add or remove rights or commands, under this model. [1]

> **Definition 2.6 (Protection system)**
> A protection system is a finite set of rights R and a finite set of commands C.

The rights of a protection system have no inherent semantics except for those implied by their use in commands. Commands are used to modify the configuration of a protection system. A definition of such a configuration is given below. Note that R and C are not part of the configuration of a system, reflecting the definition of a protection system.

> **Definition 2.7 (State of a protection system)**
> The state, or configuration, of a protection system is a triple $Q = (S, O, P)$, where S is the set of principals in the configuration, O is the set of objects in the configuration (with $S \subseteq O$) and P is an access matrix describing the rights each principal has to each object. Finally $P[s, o]$ represents the cell in P containing the set of rights that the principal s has to the object o (with $P[s, o] \subseteq R$). If no such rights exists we have that $P[s, o] = \emptyset$.

We can represent a configuration Q by annotating the access matrix with the associated principals and objects. Figure 2.2 shows an example of this approach in which there is a row for each principal and a column for each object.

[1] It is possible to simulate adding/removing predefined rights and commands in a system.

	Sam	Joe	Data
Sam	$\emptyset$	$\emptyset$	$\{\textbf{own}, \textbf{read}, \textbf{write}\}$
Joe	$\emptyset$	$\emptyset$	$\{\textbf{read}\}$

Figure 2.2: A representation of a protection system configuration

Transitioning from one configuration is done using commands. A command can, through a set of operations, modify each part of the configuration tuple. A command is defined as below:

Definition 2.8 (Protection system commands)
A command takes a simple form, allowing for a name (here represented as α), a condition, and a list of sequentially executed operations. Let X_i be principals and objects such that $X_i \in O \supseteq S$ and specifically $X_{s_i} \in S$ and $X_{o_i} \in O$. Then a command is defined as:

```
command $\alpha(X_1, X_2, \cdots, X_k)$
    if $r_1 \in P[X_{s_1}, X_{o_1}] \wedge r_2 \in P[X_{s_2}, X_{o_2}] \wedge \ldots r_m \in P[X_{s_m}, X_{o_m}]$ then
        $operation_1$
        $operation_2$
        $\ldots$
        $operation_n$
    end
end
```

Operations

The model described by Harrison, Ruzzo, and Ullman [20] is set to be as simple as conceivably possible. Because of this only the list of operations in Figure 2.3 are allowed in commands. No general purpose computation is directly possible using the proposed model as it does not reflect the protection aspect of the commands semantics.

The semantics of the available operations are formally specified in Harrison, Ruzzo, and Ullman [20, p. 463]. The operations and their associated semantics are quite self explanatory, performing simple updates of the access matrix. Because of this they are not included here.

enter r **into** $P[X_s, X_o]$	**delete** r **from** $P[X_s, X_o]$
create principal X_s	**create object** X_o
destroy principal X_s	**destroy object** X_o

Figure 2.3: The six primitive operations described by [20]

The application of an operation is written as $Q \Rightarrow_{op} Q'$ and thus $Q \Rightarrow_{op^*} Q'$ represents a sequence of operations. An operation is executed by applying its changes to an access matrix resulting in a new access matrix. If, for instance, we apply the **enter** ***write*** **into** $P[Joe, Data]$ operation to the access matrix described in Figure 2.2 we would have a new configuration reflecting the entered right. The result of that operation can be seen in Figure 2.4.

	Sam	Joe	Data
Sam	$\emptyset$	$\emptyset$	{**own**, **read**, **write**}
Joe	$\emptyset$	$\emptyset$	{**read**, **write**}

Figure 2.4: The configuration from Figure 2.2 with an added write right

Executing a command is then a matter of applying all of its operations in sequence or doing nothing, depending on the result of the conditional expression associated with the command. The invocation of a command (with a set of arguments) is described as $Q \vdash_{\alpha(x_1, x_2, \cdots, x_k)} Q'$ and implies the description given above.

Should we want to give *Joe* write access (as above) we might do this through a command that will only allow the owner of the data to provide write access. An example of such a command is presented in Figure 2.5. With such a command the *write* right to an object can only be given to a principal by the owner of the object.

```
command $confer_{write}$(owner, user, content)
    if $own \in P[owner, content]$ then
        enter write into $P[user, content]$
    end
end
```

Figure 2.5: Conferring write rights to another principal [20]

2.2.2 Analysis of protection systems

Given the formal definition of a protection system we are able to ask specific questions about a given protection system. By defining a set of requirements we can ask the same questions about several protection systems in an attempt to determine which system meets most or all of these requirements. These questions could be any of the below:

- Does the system leak the right r?
- Does the system leak the right r to the principal s?
- Does the system leak the right r to anyone other than principal s_1, s_2, or s_3?

It should be noted that all these questions are answered on the basis of an initial configuration Q. In other words; we are not interested in whether or not a right is leaked from an unattainable configuration. Additionally, we might be interested in determining if a right can be leaked given a specific configuration, effectively deciding what effects a specific right will introduce in the system.

It should be noted that a leak (given in the definition below) is not necessarily a negative. Certain rights we will want the system to leak in a specific way. We are both interested in ensuring that certain rights are not leaked and that certain other rights are.

> **Definition 2.9 (Rights leakage)**
> A command α leaks the right r from configuration Q if, after running α on Q, r is entered into a cell in the access matrix where it did not exist before running α. A protection system leaks the right r given an initial configuration Q if there exists a command in the system that leaks r from Q, or there exists a configuration Q' which is reachable from the initial configuration such that there exists a command in the system that leaks r from Q'.

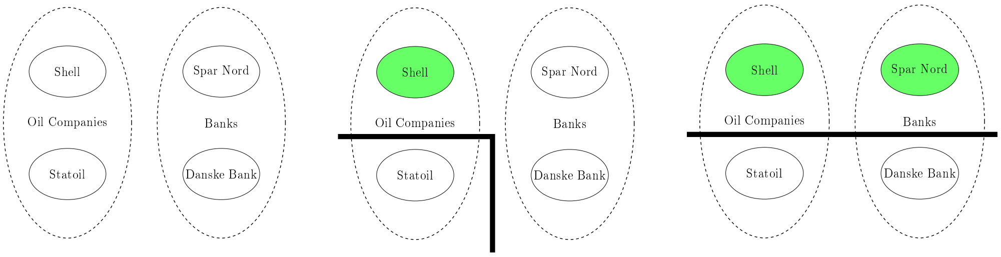
Determine if a System Leaks

We can translate the latter two types of questions, presented above, into the first one. Because of this it is sufficient to provide an algorithm that will answer this type of question.

> **Definition 2.10 (No general purpose algorithm)**
> There exists no general purpose algorithm for determining if a protection system leaks a right r from configuration Q. [20]

This is the major result described by Harrison, Ruzzo, and Ullman [20]. There does, however, exist how such an algorithm for a very simple class of protection systems called mono-operational (a system of commands with only a single operation each). Thus an algorithm exists for very simple system and not for complex systems.

In order to answer questions about specific protection systems, such as those in this chapter, custom algorithms must be defined. These algorithms must be built such that they can be applied to the protection systems being tested. The article does however not describe how to built such an algorithm.



(a) Four companies **(b)** First choice **(c)** Second choice

Figure 2.6: The Chinese Wall security policy

2.3 The Chinese Wall Security Policy

The first security models that emerged were mainly concerned with military, and were thus tailored to fit the needs of the military. However, it was shown by Clark and Wilson [7] that the current military security models were insufficient in regards to solving common commercial problems, where integrity is often of more importance than authorization. Building on top of Clark and Wilsons results in 1989 is Brewer and Nash [6], *The Chinese Wall*, which handles the problem of an analyst that must uphold the confidentiality of his clients. This means that he cannot advise a corporation if he has insider knowledge of a competitor.

The concept of the Chinese Wall security policy is illustrated in Figure 2.6. To demonstrate the purpose of the model an example of its use is given in Example 1.

Example 1 (Conflict classes)
Some firm has information about several companies; two oil companies and two banks. In Figure 2.6a this is illustrated as two groups of two companies, each group framed by a dotted oval. These groups are in the Chinese Wall security policy known as conflict of interest classes. When an analyst chooses to work with information about a company, he is not able to access information about other companies in the same conflict of interest class.

1. The analyst chooses Shell from the leftmost conflict of interest class. Because of this a conceptual wall is placed around the other company in the same conflict of interest class as depicted in Figure 2.6b.

2. The analyst chooses Spar Nord from the remaining conflict of interest class. Now the wall shields the analyst from the rest of the Bank conflict of interest class as well. This can be seen in Figure 2.6c.

The Chinese Wall Model

As exemplified in Example 1, a Chinese wall is defined as the separation between what can be accessed and what cannot be accessed by a user of the system. Information is stored in a hierarchy with three levels. This hierarchy is depicted in Figure 2.7.

- The lowest level contains individual items of information, stored in objects.
- The middle level contains company datasets that group all information that concern one company.
- The highest level contains conflict of interest classes, which group companies that are in competition.

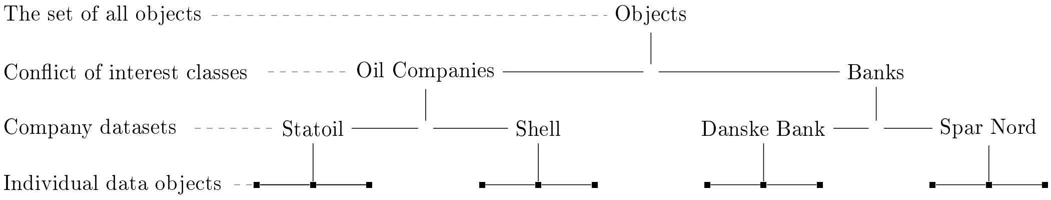


Figure 2.7: The hierarchy of the Chinese Wall model.

2.3.1 The security policy

The concept of the Chinese wall security policy is that a principal, that has access to information about company A – which resides in a conflict of interest class C, cannot access information about company B, if B is also in conflict of interest class C.

In Figure 2.7, one conflict of interest class is *Oil companies.* This conflict class contains *Statoil* and *Shell. Shell* contains objects of information that could compromise the company if *Statoil* would obtain the information. An analyst concerned with *Shell* should not be able to read any information about *Statoil* in order to uphold confidentiality, as was also described in the Example 1.

2.3.2 Chinese wall properties

This policy is being upheld by two properties. These correspond to the two identical properties defined by Bell-LaPadula, described in Section 2.1.4.

Simple Security When a principal S requests to read an object O, this can only be permitted if one of the following requirements are fulfilled:

- O is in the same company dataset as an object already accessed by S (the object is within the wall).
- O belongs to a different interest class than any of those S has previously accessed.

***-Property** When a principal S requests to write to an object O in dataset D it is only permitted if the following requirements are both fulfilled.

- Access to O is permitted by the simple security rule.
- No objects from another object O' in another company dataset D', which contains unsanitized information, can be read when requesting write access to O.

This is similar to the BLP *-property. However, here we are trying to prevent principal S from sharing D's *unsanitized information*[2] with principal S', who could possibly be unable to access D's company dataset directly, as per the simple security rule – i.e. S' is already handling another company D' within the same conflict of interest class. Even though unsanitized information cannot leave the company dataset, it is still possible to sanitize the data in order to compare companies.

2.4 Decentralized Label Model

The Decentralized Label Model (DLM) is a modern information flow control model devised in 1999 [27]. As the name suggests, it revolves around labels (similar to previous security models), however, DLM is decentralized. This means that it can be applied to systems with no trusted third party, or even any trust throughout the system. By attaching security labels to objects in code, it can be controlled how information should be shared throughout the code and at code-endpoints (input/output to/from other programs). It is possible to do both static and run-time checking of labels. The final major point of DLM is that it is formalized and even though no implementation is supplied, it should be applicable to existing programming languages.

2.4.1 Labels

A value is associated with a label. The label is a set of privacy policies. When the value flows through the system, all the policies need to be obeyed. This means that the set of principals that are able to read the value is the intersection of all policies in a label. This is known as the effective reader set.

A privacy policy is represented as an owner of some value and a set of readers. The syntax is: $<$*owner*$>$: $<$*reader*$>$. The readers are the principals allowed by the owner. The owner is implicitly allowed to read his own data. If we want a principal p that should not allow any other readers we provide an empty reader list: p: .

If we have policy K and label L, we have following notation:

- $K \in L$ – policy K is a part of label L
- $\mathbf{o}(K)$ – the owner of policy K
- $\mathbf{r}(K)$ – the set of readers of policy K

[2] Unsanitized information is any data that could cause a breach of privacy.

Labelled values are only released by the consensus of all the owners and can only be read by the readers. If one or several privacy policies are added to a label it restricts the access to the value. A label with no policies means that all readers are allowed.

If a principal is not among the owners of a label, it is the same as if it was added as a privacy policy with all possible readers, implicitly indicating that this owner has no preference of who reads the value.

Example 2 (Redundant owner)
In a system with owners o_1, o_2, o_3 and readers r_1, r_2, r_3, the following labels are defined:

$$L_1 = \{o_1\colon\ r_1, r_2;\ o_2\colon\ r_2, r_3\}$$

$$L_2 = \{o_1\colon\ r_1, r_2;\ o_2\colon\ r_2, r_3;\ o_3\colon\ r_1, r_2, r_3\}$$

Since the policy of o_3 is to allow everyone to read it introduces no restriction to the label. Therefore we say that the effective reader set of the two labels are equal.

A label can contain several privacy policies for the same principal. These are enforced just as other policies.

When examining labels, it is interesting to know if one label is at least as restrictive as another label. A label has this property if it enforces all the policies/restrictions of the other label. Formally this can be defined as:

Definition 2.11 (Label restrictiveness)
Let L_1 and L_2 be labels. Then if

$$\forall j \in L_1 : \exists k \in L_2 : o(j) = o(k) \wedge r(j) \supseteq r(k)$$

L_2 is at least as restrictive as L_1, and we write $L_1 \sqsubseteq L_2$.

This relation is applied throughout the following.

2.4.2 Rules

This section specifies a set of rules that apply when manipulating labels to avoid information leakage. The modification of labels is what makes it possible to define how data flows through a system.

Label join When deriving a value from two values, the label L_{12} of the derived value must reflect the labels L_1 and L_2 of its sources, being at least as restrictive as both of them. This is achieved by taking the union of the two labels.

Definition 2.12 (Label join)
The join of two labels L_1 and L_2, denoted as $L_{12} = L_1 \sqcup L_2$, is defined as

$$L_1 \sqcup L_2 = L_1 \cup L_2$$

Note that, since $L_1 \subseteq L_{12}$ and $L_2 \subseteq L_{12}$, we have that $L_1 \sqsubseteq L_{12}$ and $L_2 \sqsubseteq L_{12}$.

The join rule will ensure that any joining of values will also result in a joining of labels, so that any combination of rules will be upheld.

Relabelling by declassification The owners of the labels control their data, but sometimes policies are overly restrictive and one wants to relax them. This can be used to sanitize values whose security policies, in respect to one ore more owners, have changed throughout the run of a program. This works in opposition to restricting rules, which is used to ensure information is not leaked, whereas this is enabling intended leaking (of sanitized information).

The authority is a set of principals which is the authority of the process of declassification. If a process has the authority of a principal, the actions of the process are permitted. This means that if a principal is in the authority set this can be declassified.

Definition 2.13 (Relabelling by declassification)
Let $L_A = \{p_0 : , p_1 : , \cdots , p_n : \}$ be the authority label with principals $p_0, p_1, \cdots , p_n$. Let L_1 and L_2 be labels. Then if

$$L_1 \sqsubseteq (L_2 \sqcup L_A)$$

L_1 may be declassified to L_2.

Relabelling by Restriction When a value is read from a variable it has the same label as the variable. When a value is stored in a variable the label of the value is forgotten.

To illustrate the process of relabelling, the following example describes a simple assignment from one variable to another.

Example 3 (Assignment)
This example describes two variables $a = 4$ and $b = ...$ We will disregard the current value of b and focus on the relabelling that takes place when processing

$$b = a$$

> First we read the value of a and assign to it the label of a. Afterwards the value is copied to b. This copy is assigned the label of b.

To avoid leakage the label of the variable must be at least as restrictive as the value being assigned to it. This means that in the example above b is required to be at least as restrictive as a.

2.4.3 Acts-for relation

A principal in DLM can represent a user in the system, a group of users, or roles. As labels on slots are immutable, in order to have more dynamic security policies we have that principals are able to *act for* other principals. This means that a given principal in effect has every right that the principal that it acts for has.

This is also called a principal hierarchy. Formally, we use the $\succeq$-operator when representing acts for[3] such that $x \succeq y$ declares that x acts for y. A principal hierarchy P is a set of ordered pairs of principals $\{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$. This describes relations where $x_1 \succeq y_1, x_2 \succeq y_2, \cdots, x_n \succeq y_n$.

Below are a couple of examples of acts-for relations:

> **Example 4 (Acts-for relation)**
> Bob can have Amy act on his behalf by providing the acts for relation: $Amy \succeq Bob$. This implies that Amy can *act for* Bob and that any security policies that apply to Bob will in effect also apply to Amy.

> **Example 5 (Acts-for relation in groups)**
> Given a principal *Admin* that have certain system access we can represent roles/group members using the acts-for relation. The relation $Amy \succeq Admin$, signifies that Amy is an administrator and therefore can act on behalf of the administrator group.

2.4.4 Channels

The model contains two types of channels; input and an output channels. These channels provide the means of communication with outside systems.

As information can be leaked through these channels, therefore they have a label associated to them. When a value enters the system through an input channel, the value gets the label of the input channel. If a value is written to an output channel it can only be done if the label of the output channel is at least as restrictive as the label of the value.

[3] $\succeq$ is reflexive and transitive and not anti-symmetric.

2.4.5 Smart meter system example

To demonstrate how the DLM could be used in a smart meter context, we provide an example below. The example describes a smart meter system while identifying the various principals (actors) and labelled objects. This is similar to the provided motivating examples presented in Myers and Liskov [28].

In Figure 2.8 can be seen the abstract smart meter system, with principals (ovals), labelled objects (rectangles), and trusted agent (double oval). A trusted agent is a principal that is trusted by the owner of the data to modify the data and associated policies. Arrows display how information flows. There are two overall components to this system: the Data Hub and the Smart Meter. The rectangles used for these components signify only a grouping of objects.

Initially, consumers own the data contained in their smart meter, most importantly the raw consumption data. PE extracts the consumption data and transforms it into a granularity that does not reveal information about the consumer. In order for the distributor and electrical company to carry out their tasks, balancing the electrical grid and correctly billing, they have access to the transformed power consumption data. Inside the Data Hub, the distributor receives ownership of all consumption data[4], but for the power consumption data for each individual consumer C_i the only allowed additional reader is the current electrical company of C_i: EC_j. In case C_i changes to a new electrical company EC_k, the read access must be removed from EC_i and given to EC_k.

The different electrical companies will also provide their current tariffs to the data hub, so that they can be sent to the smart meters and for anyone to see the pricing of available electrical companies in case they want to switch supplier. The label is empty so that anyone can read the tariffs.

2.5 Applicability in a Smart Meter Context

Having described the four security models, a brief discussion of their applicability to the smart meter context model follows below. The limitations of each model will be described and a possible *best fit* model will be indicated.

Given the context described in Chapter 1, privacy is an important factor when evaluating the models. In order to apply a model to the context, it must thus be possible to describe which data can be sent between the various actors. This to such a degree that each actor is able to read **only** the granularity of data that they are permitted.

2.5.1 The Bell LaPadula Security Model

The Bell LaPadula security model works with classifications and categories because it is made for multiple-tier systems. Our system can be described by classifications

[4] This is actually unclear, but not important to discuss at this point

of different granularities; the consumer is cleared to view data of a fine granularity, while the electrical company can only view a coarser granularity – that just makes it possible for them to compute a bill. The distributor will retrieve a fine-grained granularity, but as an aggregate of the power consumption in an larger area. This information is required in order to understand the power distribution in the network.

The smart meter handles different types of data, consumption, control of appliances, and connected devices. These could each be described by a category that determines if the data concerns outside parties.

2.5.2 Protection in Operating Systems

Protection in Operating Systems is occupied with a different matter, namely creating a unified model that can be used to describe and possibly compare security models. It can thus be a tool for verifying that security model does not leak any data that it is not supposed to.

When a smart meter system is to be designed, using either of the security models described here, the designs can be described using the model described by Harrison, Ruzzo, and Ullman [20]. This will allow multiple ideas to be represented using the same model and go through the same type of testing for leakage.

It should however be noted that when the system has been modelled an algorithm to compare the system must still be developed. This might not be possible, but is dependent on the systems modelled. In turn, this could lead to the inverse dependency such that the model is designed in a way that a developed algorithm applies.

2.5.3 The Chinese Wall Security Policy

The Chinese Wall Security Policy contrasts the Bell LaPadula model by putting data in conflict classes. Security is achieved by placing a “wall” separating actors who access some data in a conflict class from the remaining data in that conflict class.

As the model provides a *free* choice of data in a conflict class, it will not allow us to model certain data as being available to specific actors only. Our problem is concerned with protecting data of a certain granularity from actors who do not need that particular granularity. Modelling our system in this way then seems illogical, because we cannot set up conflict classes that represent the access restrictions we require.

Because of this the Chinese Wall Security Policy will be regarded unfit for our purposes.

2.5.4 Decentralized Label Model

The Decentralized Label Model describes a way to control how data flows through a system, by attaching labels to data. This method for handling data access fits our problem very well. Data can travel from the consumer to a Data Hub and from there on to a multitude of actors from the Data Hub.

Being able to describe how data flows with varying granularity makes this model very attractive for our purposes. An example presenting an attempt at modelling the system was presented in Section 2.4.5. Note, however, that this approach requires that the model covers the implementation of both smart meters and the Data Hub.

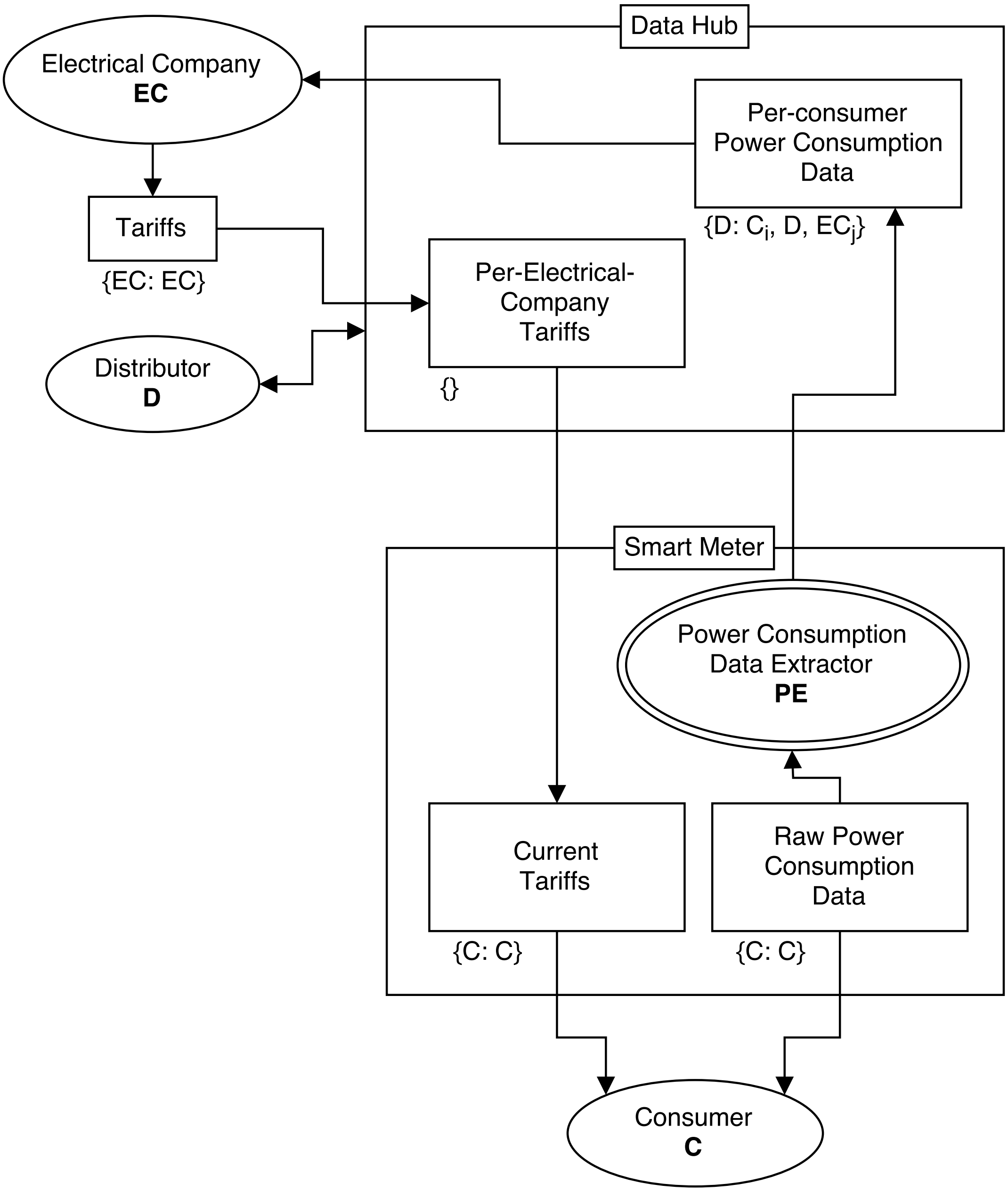


Figure 2.8: Abstract smart meter system with DLM principals and labelled objects

Chapter 3

Attack Trees

As a method of identifying security problems within smart meter systems, we have constructed attack trees for each of the attacks that involves the smart meter. First, we introduce the concept of attack trees and how we will base these on the context model defined in Section 1.3. Then we will, one by one, present the identified attacks and the overall ideas behind these attacks. A more technical description of the individual attacks will be presented in Chapter 4.

Based on the model of the system describing the relations between actors and devices, a brainstorming session in the project group has led to a set of possible attacks on the system. In Figure 3.1 the result of this brainstorming session is visualized. Figure 1.5 has been used as a basis and all the possible attacks have been marked with an arrow. An arrow pointing from actor A to actor B indicates a possible attack by A on B.

From this representation of the attacks it is evident that the consumer is threatened by the largest number of attackers. Meanwhile, he only has a single actor to perform an attack on, namely, the electrical company. This attack is described in Section 3.1.

The consumer is not found to have an attack on the power distributor (or vice-versa) as the price of transporting electricity is governmentally regulated. Additionally, it is worth mentioning that there are no attacks that do not involve the consumer.

Definition of an attack tree Attack trees consist of a set of nodes representing steps in a possible attack. The root node being the goal and leaf nodes being different ways to achieve this goal. Inner nodes are represented as logic gates such that they do not represent a possible attack themselves, but rather a rule describing how their children should be aggregated. Thus, a node represented by an AND gate would require all its child-nodes to be carried out successfully for the node itself to be carried out successfully.

To describe an attack tree, for instance how feasible it is, nodes can have different values corresponding to different variables (Boolean and continuous). For instance, a variable could be *cost* and the values assigned to it could be in dollars. Attack trees can show you which attacks are possible for which attackers. For instance, an

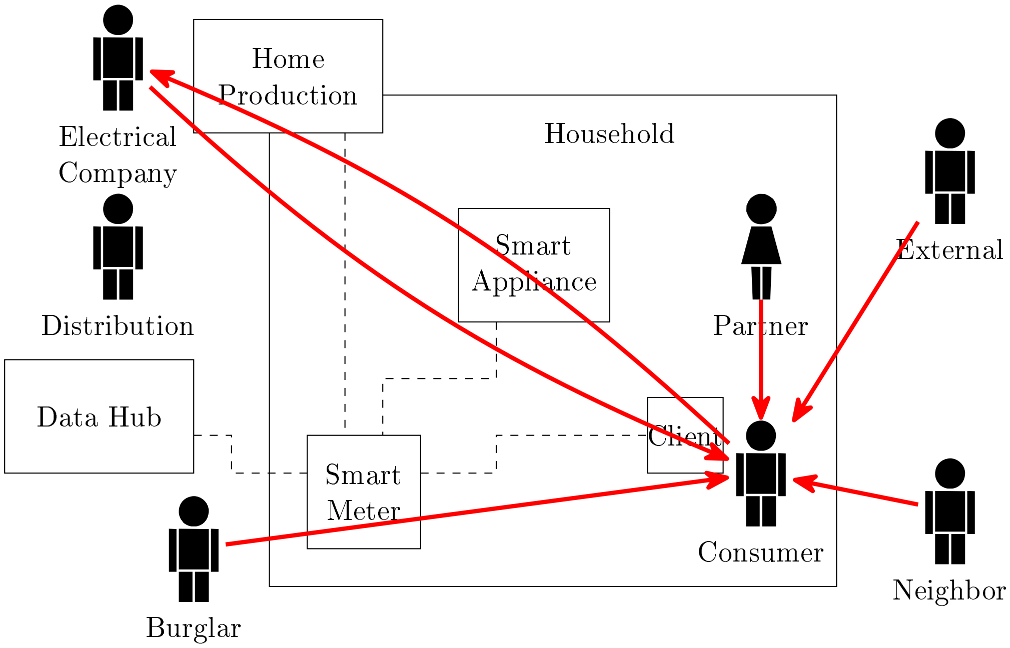


Figure 3.1: The routes of attacks in the context model

attacker with a lot of money has the possibility to do all the expensive attacks where a poor attacker has not. Also, if one builds a system, where the data that is getting protected is not very important, it might only be important to defend against cheap attacks. Making attack trees takes a lot of time and effort as one has to study the security measures in place, the possible attacks, and the possible attackers and outline this information in one or more attack tree(s).[34]

3.1 Consumer Attacking the Electrical Company

This section will contain the attack where the consumer wants to deceive the electrical company into billing for a lower amount than actually consumed. To facilitate the reader in understanding the details of the attack tree in Figure 3.2, its nodes and subtrees will be described below. In order for an electrical company to bill the consumer, he must first receive information from the smart meter about the consumer's power consumption.

The goal of the consumer is to report at lower power consumption than what is actually measured by the smart meter. The attack tree describes two different possible attacks that could achieve this goal.

3.1.1 File false complaint against electrical company

This attack involves the attacker attempting to prove that the electrical company is billing him for a too large power consumption. This despite the bill actually corresponding with correct usage and price information.

The project group suggest two different approaches to achieving this goal:

1. The user develops an application that mimics an official app, displaying the power consumption. This application will display an incorrectly low information about the consumers power consumption.

2. The user successfully alters pricing information as received from the electrical company. He will want to lower the price locally, such that when he examines his "expected bill" the total price will be lower than that billed by the electrical company. The attack requires that the consumer is able to modify the prices, e.g. by tampering incoming network packages – as pictured, as they are received by his smart meter. Network attacks will be described in Section 4.3.

Both of the above require the user to claim wrong-doing on behalf of the electrical company. Any one individual might have a problem in assuring the validity of their claim. However, applying this attack on a larger scale will provide a possibly stronger case.

Additionally, this type of attack might be carried out by the attacker on other consumers' smart meters, effectively hiding his intent by having multiple unaware consumers complain as well.

3.1.2 Falsify sent consumption

As the attacker (the consumer) is interested in what is reported to his electrical company, he could falsify this information.

In the attack tree, there can be seen four overall methods of achieving this:

1. Tampering data sent from/to the smart meter.

2. Reverse engineering the firmware.

3. Modifying the firmware in order to change how the consumer's power consumption is stored.

4. Modifying the firmware in order to change how the consumer's power consumption is sent.

Tampering

The goal of the first attack is to intercept the packages sent from the meter, modify them, and resend them to the electrical company. Alternatively the attacker might intercept packages at times where he knows his power consumption to be low and resend them at a later point in time where he knows them to be high. This is known as a replay attack (see Section 4.3.1).

Safeguards against this type of tampering is encryption and authentication. These concepts will be discussed in Section 4.2. Even though the communication between the smart meter and the distributor is both encrypted and authenticated, nothing is guaranteed to be secure. As discussed in Ferguson, Schneier, and Kohno [17], cryptography is very difficult, and protocols that are regarded as secure may contain a small error that is discovered after several years. Also, the implementation of a cryptographic system determines how secure it is, as well as the surroundings of the system [17]. If the surrounding system is leaking left and right, a correctly implemented encryption algorithm cannot be blamed.

Reverse engineering

By reverse engineering the firmware, the consumer will possibly get information that enables him to falsify the information sent from the smart meter. By performing a side-channel attack, the consumer could be able to acquire the key(s) which is used to ensure authentication and integrity of sent messages. This way the consumer could send packages which seem authentic. Side channel attacks will be discussed in Section 4.4.

Modifying firmware

The final attack(s) we propose include tampering with the smart meter's firmware. If the consumer is able to change or replace the firmware on the smart meter, he could be able to modify the way that the smart meter either stores or sends power consumption data. Tampering with the firmware can happen in a number of ways and is a broad topic on its own. This will not be further elaborated in this report.

3.2 Burglar Attacking the Consumer

This section provides an analysis of an attack in which a burglar wants to break in to the house of the consumer and steal his valuables. Several options to achieve this goal are explored. These are listed below:

- Figure 3.4: Compromising the security on the smart meter.
- Figure 3.5: Compromising the device (the client) that the consumer is using to communicate with the smart meter.
- Figure 3.6: Installing an ECM (see Section 4.9) and monitoring the consumer.
- Figure 3.3: Performing a *"physical"* attack, involving no technology.

Representing the attacks as four separate attacks might be somewhat misleading. For instance, the burglar might choose a physical means of determining if the consumer is at home, even though he initially compromised the smart meter. Thus, the four attack trees might be combined to properly describe the full spectrum of the burglar's attack strategies.

The trees are represented as four trees here to show the correlation between how the burglar gains access to the system and the possible attacks this makes available. The separated trees provide a clear visualization of this relation.

This representation is used to build a model of the possible threat to the consumer. As this representation fails to fully represent attacks that employ multiple strategies, special care should be taken when modelling the trees. Using a model that joins the four trees will allow for a comparison of the validity of each device, as an entry point for an attack.

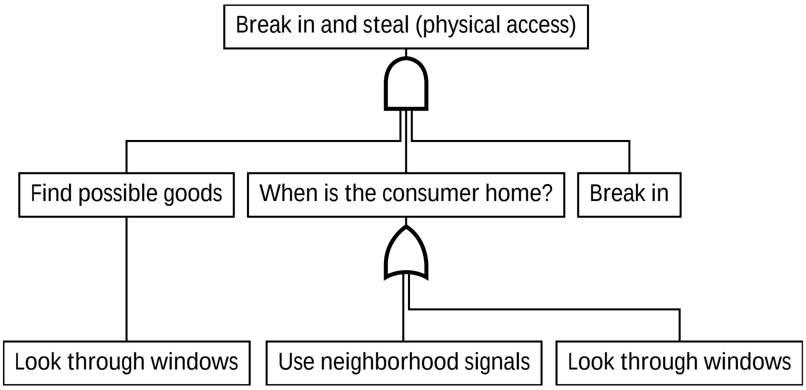


Figure 3.3: A burglar attacking the consumer through physical means.

In the following, the various attacks on the system are described. As the attacks are similar in structure they are described in terms of their difference. The structure consists of an initial step where the burglar gains access to information. Then a step to determine which valuable items the consumer has and a step to determine if the consumer is home. The final step in the attack is stealing the consumer's valuables.

3.2.1 Gaining access

In order to access information about the power consumption of the consumer, the burglar needs to gain access to either the smart meter, a consumer device that controls the smart meter, or install an ECM unit (see Section 4.9). The following will discuss the subtrees concerning these access methods.

Compromising the smart meter

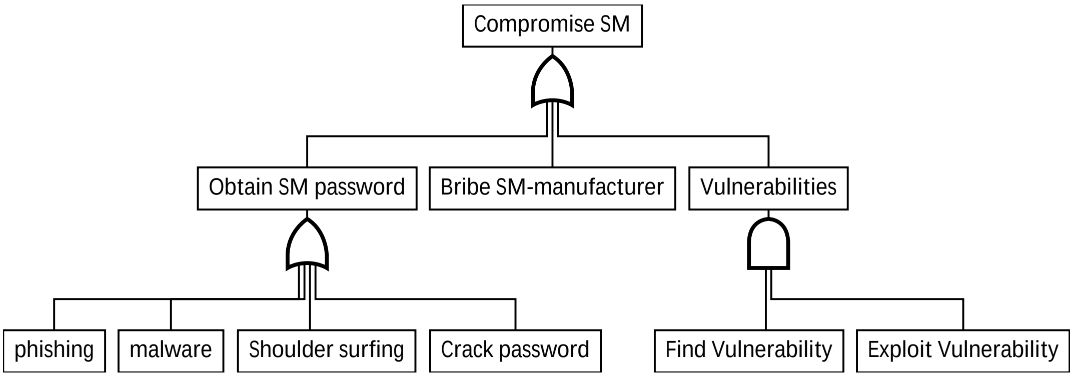


Figure 3.7: Subtree for compromising the smart meter.

There are three general ways of compromising the smart meter. The subtree can be seen in Figure 3.7. First of all, if you have the password for logging it is possible to access the same information as the consumer is. There are a couple of methods for gettings one's hands on this password.

Phishing The burglar can attempt to get the password directly from the consumer by posing as the electrical company or similar. Phishing will be discussed in Section 4.6.1

Shoulder surfing If the burglar has the possibility of watching the consumer in a context where he enters his password, he can retrieve the password, or hints about the password, by shoulder surfing. Shoulder surfing can also lead to information about the consumer, or the smart meter, that will help the consumer carrying out one of the other option, such as finding the brand of smart meter for constructing a phishing email. Shoulder surfing will be discussed in Section 4.6.2

Malware It is also possible to get the password by infecting a client device of the consumer with some malware. This piece of malware can either send the password back to the burglar or provide access to the system. Malware will be discussed in Section 4.7

Crack password If the other options do not amount to anything, the burglar can resort to cracking the password. Password cracking will be discussed in Section 4.8.

The second option is to bribe the smart meter company to make some changes to the firmware that help achieve the goal. This could be adding a backdoor to the firmware, or to change how the firmware handles consumption data. This is an effective way of achieving something on a lot of smart meters, as the smart meter manufacturer will push the firmware to a lot of customers.

Another option is to find some vulnerability in the smart meter firmware. Depending on the vulnerability, the burglar can possibly perform actions that was not intended by the manufacturer. Other vulnerabilities will make it possible to access the same information as available to the consumer. An example of a vulnerability is a buffer overflow. Buffer overflows will be discussed in Section 4.5. If a buffer overflow is found it may make it possible to run arbitrary code on the smart meter which can possibly open for everything.

Compromising the client

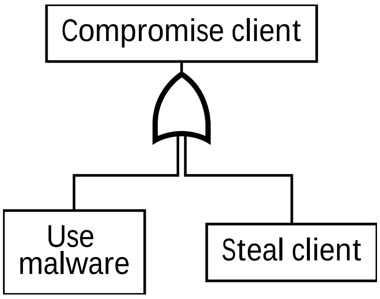


Figure 3.8: Subtree for compromising a client of the consumer.

An alternative point of attack is the client that the consumer uses to control the smart meter. The subtree for this case can be found in Figure 3.8. The obvious possibility is to steal the client that the consumer uses. If the consumer uses a tablet for controlling the smart meter, this can be done when the consumer is at out of the house or at work by “conventional” mischievous methods. Assuming the burglar can overcome the login to the client, he will be able to access the smart meter as if he was the consumer.

Alternatively, the burglar can use malware to gain either the password to the smart meter or access to the client. This step is similar to the malware item earlier in this section.

Installing an ECM

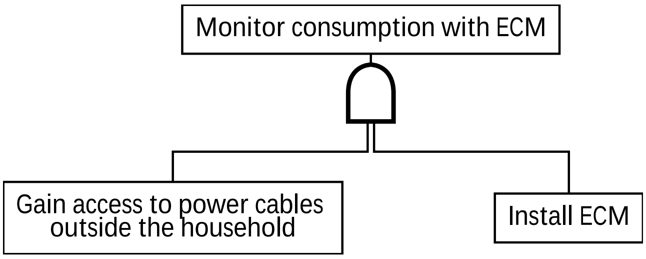

Monitor consumption with ECM
Gain access to power cables outside the household
Install ECM

Figure 3.9: Subtree for monitoring the consumer with an attached ECM.

If it is not possible for the burglar to get access to the smart meter, or the client, from the outside, he has another option. An External Consumption Monitor (ECM) is a device that can monitor the consumption of a electricity cable (ECM will be discussed further in Section 4.9). If the burglar can find the incoming power cable and attach such a device, he can get roughly the same data as the consumer gets through his smart meter and therefore he will be able to perform many of the same attacks as with access to the smart meter.

The subtree depicting how to achieve this can be seen in Figure 3.9.

3.2.2 Expensive goods

The first step for the burglar, after gaining access to the system, is to determine if anything is of value in the house. The burglar can choose any house he wants and will use the information obtained to determine which house will yield the highest total worth for the least amount of effort.

Physical means The physical method is simply to look in through windows of the house and consider what can be seen. This method has some obvious shortcomings. Not everything can be seen from the outside, and in multiple story houses several stories may be impossible to evaluate. Is is also rather inaccurate – it may be possible to see a TV, but is it the newest product from this year, or is it an older model that is virtually impossible to sell?

Compromised smart meter or router If the burglar has compromised either the smart meter or the internet router of the consumer, he will be able to find out what products are in the home. This is possible either through some smart meter database that shows what can be controlled, or through network traffic which can indicate what products are being communicated with.

Power consumption If the burglar has access to the power consumption (either smart meter or the ECM), it will be possible to detect product signatures in this data, see Section 1.2.1. This is difficult and possibly ambiguous, but it may provide pointers on what he can expect to find.

3.2.3 Is the consumer at home?

When the burglar decides on a home to be his target, he needs to plan when to break in. Before doing that he needs information about when the consumer is home.

Physical means The burglar can use some physical methods to determine if the consumer is at home. He can look through the windows to check for activity in the house. This technique is not guaranteed to show that nobody is home, but if there is activity the burglar can be sure that someone is at home. Over a longer period of time this technique can provide information about the habits and work hours of the consumer, making it easier to plan when to break in and when to be out again.

Another technique is to put indicators in the neighborhood. This could be small signs on the mailbox, a can behind the tire of a car, or a knocked over bike. The purpose of these subtle indicators is to find out if the home is empty for a longer period of time. If the consumer is home it is likely that he will erect the bike and/or remove the mark from the mailbox. If all indicators are left untouched it is likely that the consumer is not at home. This technique is very fast to apply to a consumer's home and can thus easily be applied to multiple homes.

Compromised smart meter Having access to the smart meter may gain access to schedule information which can be an indicator of when the consumer is at home. If the consumer schedules laundering during the night it may indicate that he plans to be home shortly after after the laundering has ended. The schedule may also provide information about work hours of the consumer, making it possible for the burglar to plan the break in.

Having control over the smart meter also provides the power to turn devices on or off. The burglar can use this to reinforce any suspicion he has that the consumer is home or not. If he thinks that the TV is turned on by some timer mechanism he can force it to turn off. If the consumer is really at home he will probably turn it on again. Provided that this action must be done manually, this is a sure-fire proof that the consumer is at home. Another test could be to turn the stereo on and put the volume to an abnormally high level. If the consumer does not react to this, he is either sleeping very tight or not at home.

Power consumption If the burglar has access to the smart meter or an ECM (External Consumption Meter) he has some options that can help him determine if the consumer is at home.

From the power consumption it is possible to determine which devices are currently on. If the TV is turned on and off in irregular intervals (timers exist that can turn devices on and off at predetermined intervals), it is an indicator that the consumer is home.

3.2.4 Break in

When the burglar feels confident that he wants to break in to the house, he can use the smart meter for some additional help. The smart meter can provide information about the security status of the house.

If there is a security alarm or camera, this will be visible on the smart meter by the same techniques mentioned in the "Expensive goods" subtree – either in the consumption data, network data, or smart meter database. If the consumer has some security devices installed, the burglar can turn them off entirely and make the break in easier before he even enters the property.

The last step is to steal all the expensive goods in the house of the consumer. The smart meter (as well as surveillance) might provide the burglar with information that lets him know when the consumer is usually home, and he therefore knows when to be out in order to not get caught.

3.3 Neighbor Attacking the Consumer

If the consumer has some noisy habits, the neighbor may become irritated on his practices. Gaining access to the smart meter will make it possible to either shut off devices completely or control the devices. That way he can change the schedules to a better point in time (according to him) or turn down the volume of some device.

The attack tree describing the attack can be found on Figure 3.10. The subtree describing how to get access to the smart meter is identical to that of the burglar, which was described in more detail in Section 3.2.1.

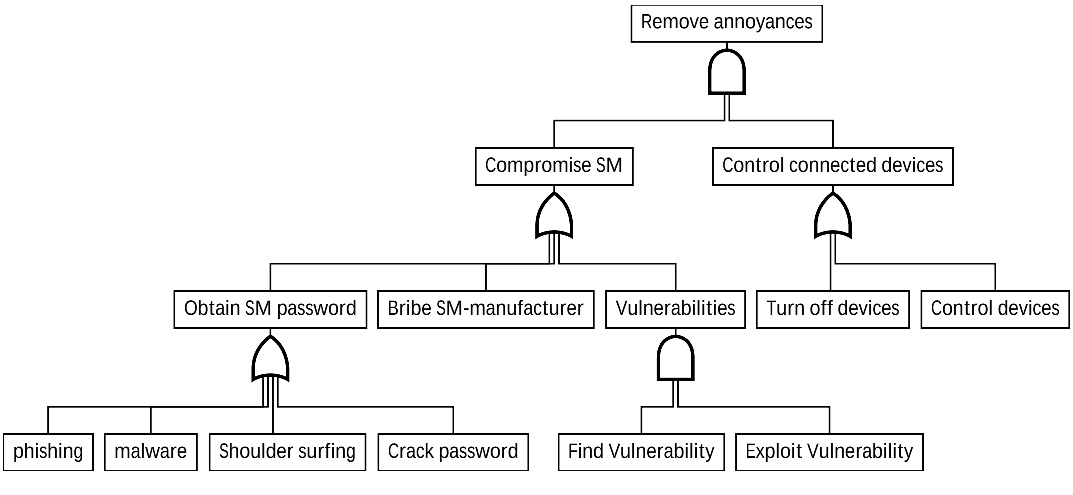


Figure 3.10: The neighbor attacking the consumer by turning off or controlling his devices.

3.4 Partner Attacking the Consumer

The partner is an actor with relation to the consumer, but, unlike the other actors, partners usually can get away with more misdoings inside the consumer's property or house.

With this attack, the partner wishes to monitor the consumer, possibly with following revenge – if monitoring leads to the conclusion that the consumer was cheating on the partner. What distinguishes the partner's attacks from the burglar's and neighbor's is that the partner has increased access to the consumer's property, including the smart meter and related objects.

The attack trees are separated into the four modes of access for the same reasons as the burglar was (see Section 3.2). The attack trees are listed below:

- Figure 3.14: Compromising the security on the smart meter.
- Figure 3.13: Compromising or extracting data from the device (the client) the consumer is using to communicate with the smart meter.
- Figure 3.12: Installing an ECM and monitoring the consumer.
- Figure 3.11: Performing a *"physical"* attack, involving no technology.

Each tree represents the possible attacks and gains, depending on the kind of access acquired. There are two or three overall steps to the partner's attack plan: *Gain access, Surveil, Get revenge*, which are repeated in the attack trees.

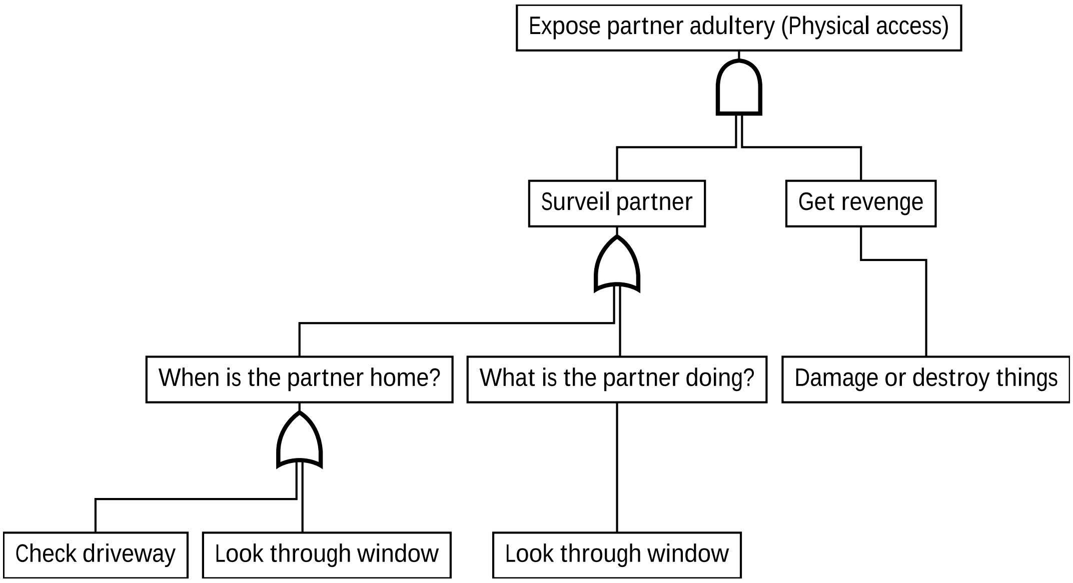


Figure 3.11: The Partner attacking the Consumer by physical means.

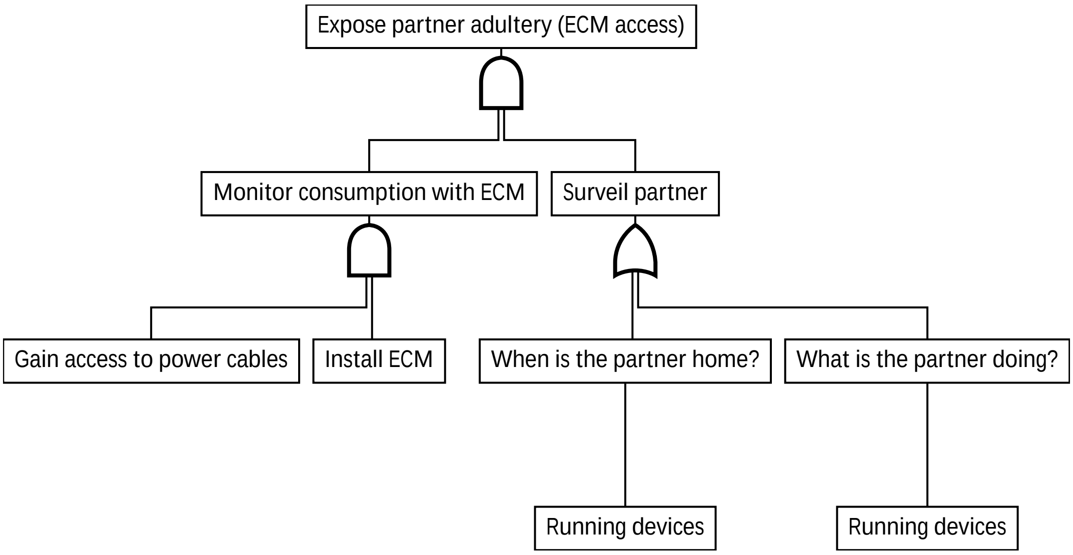

Expose partner adultery (ECM access)
Monitor consumption with ECM
Surveil partner
Gain access to power cables
Install ECM
When is the partner home?
What is the partner doing?
Running devices
Running devices

Figure 3.12: The Partner attacking the Consumer using an ECM.

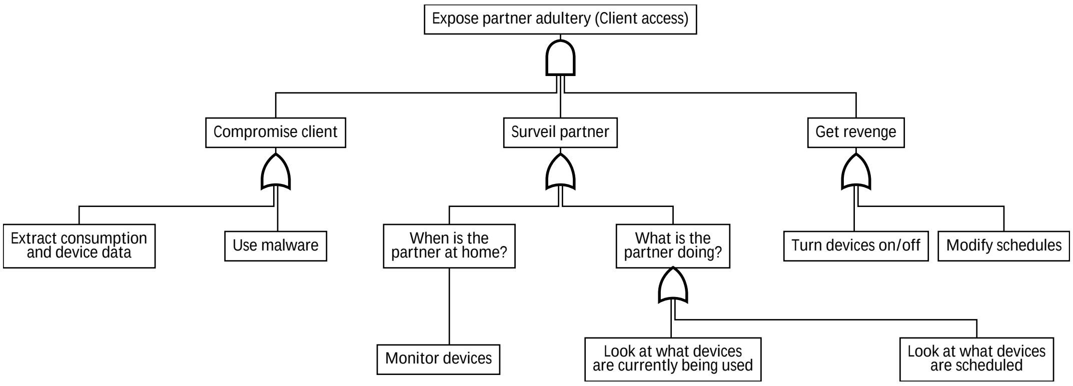

Expose partner adultery (Client access)
Compromise client
Surveil partner
Get revenge
Extract consumption and device data
Use malware
When is the partner at home?
What is the partner doing?
Turn devices on/off
Modify schedules
Monitor devices
Look at what devices are currently being used
Look at what devices are scheduled

Figure 3.13: The Partner attacking the Consumer by compromising his client.

3.4.1 Access

First of all the partner needs to gain access that can give the partner valuable insight into the doings of the consumer. In relation to power consumption, the partner has three ways of accessing the consumer's data, which could give the partner the details the partner needs to determine whether the consumer is home or not.

Smart meter

The first way is to compromise the smart meter. This variation of the attack is identical to that of the burglar which was described in Section 3.2.1.

Client

The other option is to gain access to the client that the consumer uses to control the smart meter. This variation is similiar to the attack of the burglar described in Section 3.2.1, with the exception that the partner has easier access to the client device. A situation may arise where the partner has access to the client, and can then copy data from the device.

ECM

The partner could install an ECM on the power grid in the house, giving the partner direct power consumption data. This attack is similar to the burglar using an ECM, described in Section 3.2.1. Again, the only variation is that the partner is closer to the consumer and therefore has more opportunities for installing the ECM. For instance, the partner could install it directly on the smart meter if the smart meter is hidden in a cupboard in the basement.

3.4.2 Surveillance

The main goal of this attack is to surveil and through this surveillance expose the consumer's misdeeds. The nodes regarding devices are dependant on gained access through the *Gain access* subtree, as is represented with individual attack trees based on acquired access.

The partner's initial option is through a physical presence, such as looking through a window. However, assuming that the partner was able to somehow get hold of the consumer's power consumption data, the partner now has options to expose the consumer from a distance.

First of all, the partner could determine whether the consumer is home, and shouldn't be, or that the consumer isn't home, but should be. This can be done by looking at which devices are scheduled for tasks or from looking at the device power signatures. The same method can be used to determine what the consumer is doing if the consumer is home, such as brewing more coffee than usual or taking unusually long showers. In Section 1.2.1 it was shown how to obtain this information from consumption data.

3.4.3 Revenge

The final step of the attack plan is an optional cold dish of revenge.

The partner could of course destroy stuff by applying physical manipulation in one way or another. However, if the partner wanted to get creative, the partner could use the previously gained access to the smart meter to manipulate the consumer psychologically. The partner could turn on/shut off the consumer's appliances at inconvenient times. If the partner wanted to get even more creative, the partner could modify the appliances' schedules such as turning everything on at late night with low intervals.

3.5 Electrical Company Attacking the Consumer

The electrical company sells electricity to the consumer as a service. Some electrical companies may want to raise the earnings by charging the consumer more than they actually have consumed. In order to make it difficult for the consumer to detect the changes the electrical company needs to make the bill and the amount reported by the smart meter identical. It is therefore necessary to compromise the smart meter.

Gaining access The attack tree for this scenario can be seen in Figure 3.15. Section 3.2.1 described how to compromise the smart meter and is similarly applicable here. The electrical company has more limited possibilities than described, because logging into the smart meter does not give the electrical company any advantage in earning more money.

Altering the consumption When access is gained, the electrical company can abuse the attained power to change the actual data on the smart meter. This data is then later sent to the Data Hub and the electrical company will calculate the bill as if nothing happened.

A slightly different option is to bribe the smart meter manufacturer into changing how the smart meter stores consumption data. If the smart meter records a slightly higher value than actually measured, the electrical company will benefit greatly from rolling that firmware out to thousands of users. This method is also more transparent because the consumer will have a reported consumption that is only slightly higher than expected, and it will therefore be harder for the consumer to recognize.

3.6 Distribution Company Attacks

During our brainstorm process we have not been able to come up with any attacks from the distribution company. The distribution company is just the intermediary that manages the physical connection through the power cables, so there is no immediate attack routes through the smart meter for the distribution company.

3.7 Information Gatherer Attacking the Consumer

This section describes the attack tree for people who want to obtain private information about the consumer and use the information to construct a profile of the consumer. Some of the possible actors that want to obtain information from the consumer are listed below:

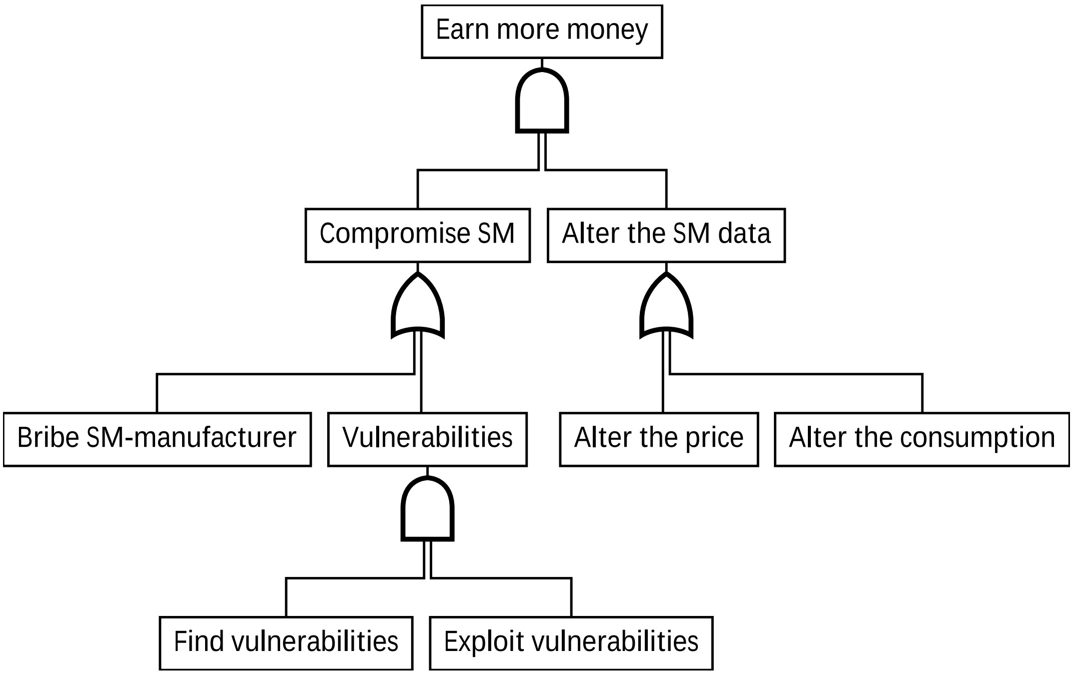


Figure 3.15: The electrical company attacking the consumer.

- Commercial company – to better aim advertisements and commercials directly to the consumer
- Appliance company – aiming advertisements and commercials as well as information about their market share and their competition
- Intelligence agency – for surveillance
- The government – wanting to monitor their citizens

The attack tree can be seen in Figure 3.16. The overall goal of the information gatherer is to construct a profile of the consumer. The "traditional" way of doing this would be to buy this information from someone who has gathered this information. This could be other actors from the list mentioned above, like Facebook or Google, who continuously get information about its customers. The introduction of the smart meter into the home of consumers provide a new way of collecting this information. The subtree describing how to profile a customer will be described in the following sections.

3.7.1 Obtain power consumption

In order to analyze the consumption data of the consumer, the information gatherer must obtain the consumption first.

Compromise smart meter

This subtree is similar to the subtree of the burglar getting access to the smart meter, which was discussed in Section 3.2.1. The only difference is that *shoulder surfing* is omitted because the information gatherer is further away from the consumer and does not have the opportunity to get close to the consumer.

Compromise client

This subtree is similar to that of the burglar described in Section 3.2.1. The only difference is the omission of the *Steal client* option, again, because the information gatherer is not in the immediate vicinity of the consumer.

Buy power consumption

Another possibility is to buy the information either from someone who has gained access to it or directly from the consumer. When buying from the consumer, social engineering can be used to convince the costumer to sell his data, possibly cheap or against his will.

3.7.2 Analyze power consumption

After gaining access to the smart meter and the consumption data, there are many things that can be revealed about the household and the consumer's activities and appliances. See Section 1.2.1 for an example of how the consumption data could be used to reveal this information.

3.8 3rd Party App Developer Attacking the Consumer

The smart meter system we modeled in Section 1.3 allows for developers creating their own apps for displaying the smart meter data, as well as control the connected appliances. This makes it possible for the developer to handle the data of the consumer and thus makes it possible for a malicious developer to send the data back to himself. The purpose of this could be to sell the information to information gatherers who are willing to buy the information as mentioned in Section 3.7.

The attack tree for the 3rd party app developer can be seen in Figure 3.16. As the app has direct access to the smart meter data, it is easy for the developer to obtain the data and send it to a database through the platform that the application is installed on. He can then choose to analyze the data and create a profile (this was discussed in Section 3.7). At last he can sell the profile (or even just the raw data) to some information gatherer.

3.9 Appliance Company Attacking the Consumer

A manufacturer of appliances can have similar objectives as the attacker in Section 3.8. The attack tree is also very similar as seen in Figure 3.18 and thus the description will not be repeated. The difference is how the data is obtained. The third party app develop has direct access to the data and the potential of sending the data through the platform that the application is installed on. An appliance company needs to obtain the data through the firmware on the appliance and send it through either a network connection on the appliance or through the smart meter. Although possible, these means are less likely than those described in Section 3.8

3.10 External Powers Attacking the Consumer

In Section 1.2.2, some more distant powers were mentioned - states, terrorist organisations, environmental activists, and blackmailers. Abusing the consumer's smart meter, for these distant powers, is a small piece in a bigger picture. They all want to shut down the power of the consumer in order to achieve a bigger objective – power, political influence, or money. Also, the government of the consumer can abuse this kind of power. The attack tree depicting this attack can be seen in Figure 3.19.

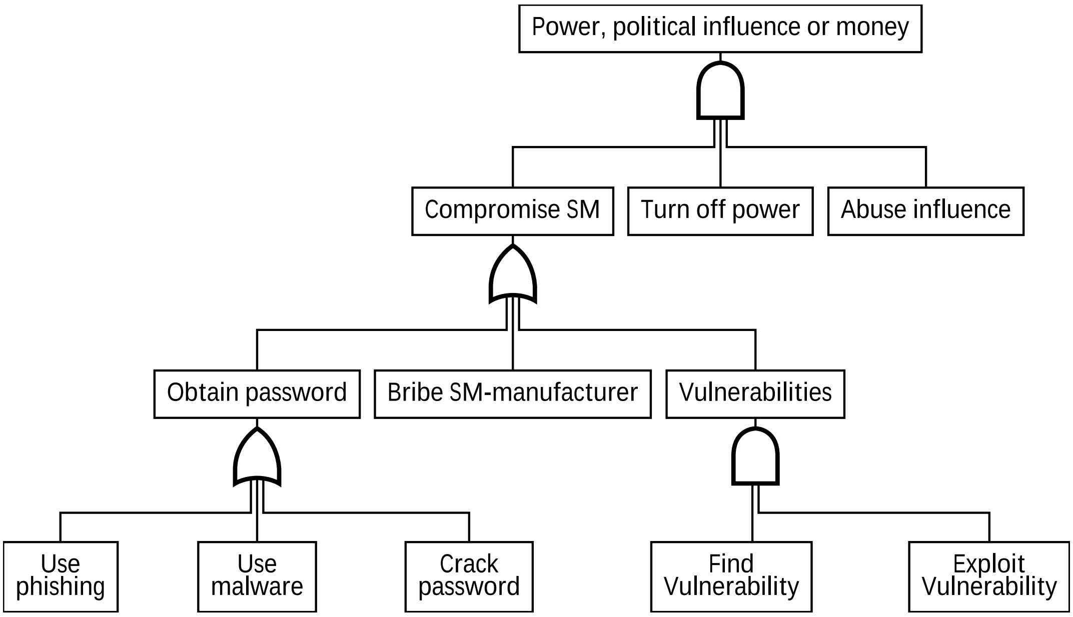


Figure 3.19: External powers attacking the consumer.

Chapter 4

Attack Techniques

In order to examine how a smart meter can be abused, we need some basic security concepts. The attacks presented here will relate to the attack trees presented in the previous chapter. First, we will go over cryptography concepts, and afterwards a selection of relevant attacks will be explained.

4.1 Terms

Alice and Bob are used to present two parties. Generally, Alice wants to send Bob a secret message. Eve is used as a third party who is eavesdropping on the communication between Alice and Bob. This typical setup can be seen in Figure 4.1.

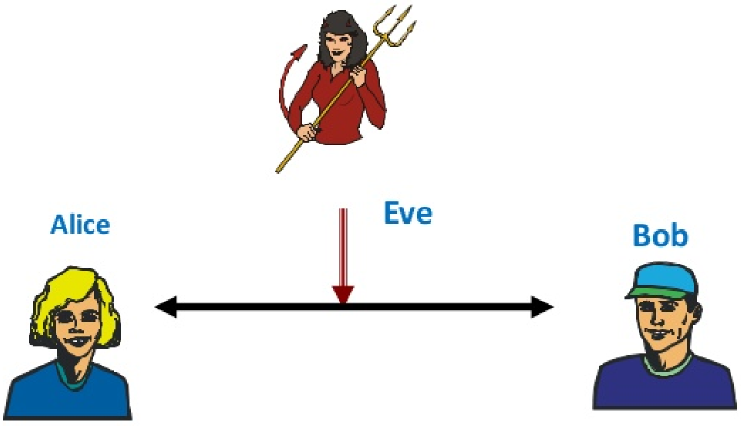


Figure 4.1: Alice, Bob and Eve.[1]

[1] `https://image.slidesharecdn.com/quantumcryptography-130913005611-phpapp01/95/quantum-cryptography-13-638.jpg?cb=1379033822`

4.2 Cryptography

This section will briefly introduce concepts from cryptography relevant to the project and will be based on [17].

4.2.1 Encryption

Encryption is a concept used for keeping the contents of messages between two parties secure.

When Alice sends a message to Bob, over some insecure channel, it is impossible to ensure that Eve does not read the message. It is therefore safe to assume that a message sent by Alice to Bob can be read by Eve. The situation can be seen on Figure 4.2.

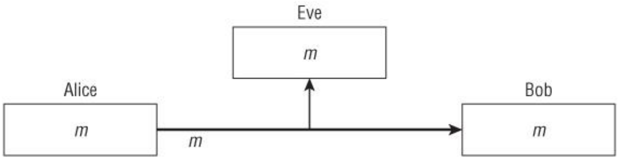


Figure 4.2: Eve easily intercepts and reads m. From Ferguson, Schneier, and Kohno [17, p. 50]

The solution is to encrypt the message m with a key K_e that is agreed on beforehand. Now, Alice can send an encrypted message to Bob which Eve will be able to read but not understand. The situation is presented on Figure 4.3.

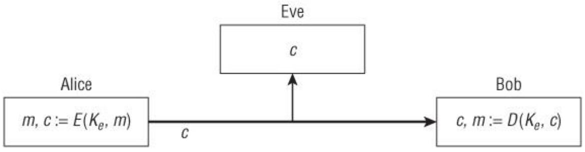


Figure 4.3: Eve can still intercept but cannot understand m. From Ferguson, Schneier, and Kohno [17, p. 50]

The encryption function is written as $c = E(K_e, m)$, where c is the resulting cipthertext. When Bob receives c he can decrypt it with the encryption function written as $m = D(K_e, c)$. Eve receives the same c, but does not have the encryption key and therefore cannot read the original cleartext from it. A good encryption further protects the cleartext by making it impossible for Eve to learn any information about m, besides the length and the time it was sent.

4.2.2 Authentication

Authentication is a concept used for ensuring that the traffic between two parties is not being tampered with.

When Alice sends messages to Bob over some insecure channel, it can be possible for Eve to change how Bob receives the messages. Eve can alter the sent message *m* to *m'* and send *m'* to Bob without Bob knowing. Eve can also change the order of messages, so that a message comes much later than intended. If Eve deletes a message, Bob will never know that this message existed. Eve can even create her own message and send it, without Bob knowing that this message was not created by Alice.

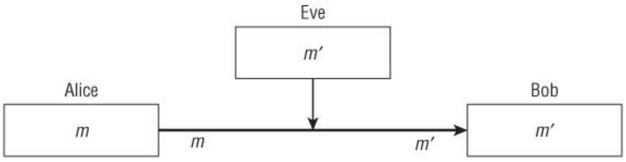


Figure 4.4: The basic problem, who sent *m'*? From Ferguson, Schneier, and Kohno [17, p. 52]

Part of the solution to these problems is authentication of messages [17, p. 52]. Alice and Bob have a shared secret key K_a for authentication. When sending message *m*, Alice first computes a Message Authentication Code (MAC). This authentication code a is calculated as $a := h(K_a, m)$, where h is the MAC function. Alice now sends both the MAC a and the message m to Bob. When Bob receives a and m he computes the MAC of m and compares it with the a he received from Alice. The situation is outlined in Figure 4.5

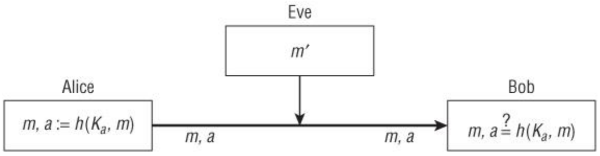


Figure 4.5: Setup for authentication. From Ferguson, Schneier, and Kohno [17, p. 53]

If Eve wants to modify the message *m* to the different message *m'*, and simply replaces *m* with *m'*, Bob will be able to tell when computing the MAC of *m'*. This is the case because of the design of the MAC function. A MAC function is designed such that two different messages *m* and *m'* are very unlikely to result in the same MAC. Bob will therefore discard message *m* as an compromised message.

Even with this countermeasure, Eve will be able to record a message and MAC pair and send it to Bob at a later point in time. She can also still delete messages, without Bob knowing that it ever existed. Because of these flaws, authentication is almost always accompanied by a numbering scheme, which makes it possible for Bob to determine if there is being tampered with the message order. Eve will still be able to entirely stop the communication between Alice and Bob, or delay it.

4.3 Network Attacks

This section will contain attacks that work by interfering with network traffic.

4.3.1 Replay attacks

This description of replay attacks is based on Ferguson, Schneier, and Kohno [17, p. 223].

Eve can perform a replay attack if she is able to record a message from the communication between Alice and Bob. She can then send the message to Bob at a later point in time.

A related concept is retries. During the run of a protocol, Alice does not receive a response from Bob. This could be because Bob did not receive Alice's request or a similar network problem. To solve this Alice sends her message again. This message is known as a retry.

Now Bob can receive both a replay from Eve and a retry from Alice. Bob needs to deal with this properly, so that Eve cannot abuse replays to her advantage. Abusing replays is known as a replay attack.

4.3.2 Packet injection

This description is inspired by Schoen [35]. If Eve can intercept communication between Alice and Bob, she can inject packages into the stream of messages without either party knowing. A packet is injected by constructing a packet with the source address as one of the endpoints – Alice or Bob. In this way it looks like Alice or Bob originated the packet, and it seems like part of the communication. By abusing this, it can be possible to affect the communication of Alice and Bob. Eve could be sending packets that simulate a protocol error, in order to disrupt the communication between Alice and Bob. Or, by injecting new packets, Eve can alter the course of the communication, which can be dangerous if the communication involves sensitive information.

4.4 Side Channel Attacks

This description of side channel attacks is based on Ferguson, Schneier, and Kohno [17, p. 132].

Side channel attacks is the class of attacks that take advantage of an alternate channel of information. This could be how much time an operation takes, the power consumption during the operation, or magnetic fields.

There are means for protecting against this type of attacks, but it is hard to eliminate information leakage from all possible channels [17, p. 132]. In the case of a lot of devices, smart meters included, the device is in the hand of the attacker. Physical access to a security device is almost impossible to protect against [17, p. 132].

Differential Power Analysis

One particular side channel attack is differential power analysis, described in Kocher, Jaffe, and Jun [21]. Here, an attack is carried out on a device that is encrypted using DES. Power traces of the device are recorded when it is performing its encryption operations. Because of the way semiconductors work, different operations will have a discernible signature on the power trace. By recording enough traces, information about the key can be derived from properties of the trace. As the amount of information increases through this method, the key can eventually be derived.

4.5 Buffer Overflow

The following is based on Foster, Osipov, and Bhalla [19, p. 18] and Ruwase and Lam [33, Section 1.1].

A buffer overflow is an attack on a vulnerable program, which modifies its memory state. A vulnerable program is a program that does not check if a given input exceeds its buffer size in memory. If the program gets some input, which exceeds its memory buffer size, the program will copy the rest of the input in some adjacent buffer. This gives the attacker the opportunity to control the machine where the program is executed, with the same rights as the program(for instance as “root”).

4.6 Social Engineering

This section will contain attacks that abuses the over-trusting, or even gullible, nature of some individuals. This individual could be providing sensitive information directly to a seemingly trustworthy, but actually malicious, party. It could also simply be revealing too much information, by using a personal device in a public setting, such as looking on your phone while sitting on a bench in an airport.

4.6.1 Phishing

The following is based on Anderson [4] and Dhamija, Tygar, and Hearst [11]. Phishing is the process of tricking a user to give up sensitive information, such as user names or passwords, by posing as a trusted party. This is typically done by using a malicious website, posing as a legitimate website that the user would normally use. This could for example happen by inserting a fake link in an email and send it to the user. The tricky part about phishing is to convince the user, that the email with the link, and possibly the link itself, is legitimate.

An example could be a fake email to get a user’s credentials for PayPal. The email would contain the logo from PayPal and would tell the user to update his account information. If the user then uses the link from the email he will get transferred to a malicious site. When the user then enters his user name and password, and submits it to the site, the attackers will have his credentials. To make it even worse for the user, the attacker will redirect the user to real PayPal website and the user will have even less of a chance to know what happened.

4.6.2 Shoulder surfing

Another technique for gaining information, is to observe a user entering the information on a client device, as described in Long [23]. This could be observing a PIN being entered at an ATM or observing the password being entered on a tablet device. Other than passwords and PINs, shoulder surfing can also reveal other information about the user. A bookmark in the browser could for instance reveal the manufacturer of the smart meter, which could be an advantage for an attacker.

Shoulder surfing in combination with bad security practices can end up circumventing even the most advanced security system. For instance if you publicly display your password, a lot of security measures are already bypassed if someone intercepts it.

4.7 Malware

The following is based on Anderson [4, p. 644]. Malware is a collective name for malicious code. This includes *trojans*, *viruses*, *worms*, and *rootkits*. Common for these are that they infect a computer of a unsuspecting person and try to achieve something harmful to either the person or his data. For example, a rootkit can provide control over the computer for the attacker, while a virus can encrypt some crucial files on the hard drive and demand ransom. A piece of malware could also install a backdoor in another piece of software, bypassing the intended security features.

In this context, it is important to note that malware can provide access to a smart meter in a number of different ways. If the password to the smart meter is stored on some client, the malware will be able to send this back to the attacker. A rootkit on a client will provide a possibility to control the smart meter through the client.

4.8 Password Cracking

Passwords are the most used authentication method, but not the safest. As users have to remember a lot of passwords, they tend to have a small collection of passwords which they use. Commonly, users just remember the collection or write it down on a piece of paper. If the user cannot remember a password, he will try all passwords, or just reset the password.[18, 5, 8]

The following is based on Marechal [24]. Passwords are generally not stored in plain text. The most used method is to cryptographically hash the passwords. Typically, a hashing function takes the plain text password as input and a *salt*. The salt is some *random* data used as additional input in hash functions to, for instance, prevent a dictionary attack[2]. Password cracking is the process of retrieving these passwords

[2] A dictionary attack is when one has a list of possible passwords and tries them all.

from hashed passwords. This process can be viewed as three steps:

1. Choose a password that is likely used by the user
2. Find the salt
3. Hash the chosen password with the salt and see if it matches the original hash

When one has cracked a password, this often gives access to other sites. Because, as mentioned earlier, users reuse the same password for different sites.

4.9 External Consumption Meter

An External Consumption Meter (ECM) is an electric meter which is attached to the incoming electricity phase(s) in the panel. It consists of a measuring unit and a device to distribute these measurements, called an energy monitor. The measuring unit is attached to the phase(s) and outputs raw measuring to the energy monitor.

The energy monitor can analyze the data and distribute them, this depends on which solution one chooses. Some ECMs have an energy monitor which distributes the consumption data outside the home network, while some also have the ability to analyze the consumption data.[10, 13, 26]

Installing an ECM can be considered an attack, because it can gather data about a person which can then be analyzed. The implications of data gathered by an ECM were discussed in Section 1.2.1.

Chapter 5

Conclusion

In this report, we gave a short overview of the coming smart grid and smart meter system and possible security pitfalls related to it. Firstly, we gave an overview of the context and the problems we found the most interesting. We then investigated existing security models and evaluated how they could assist in solving any of the problems we found in the context.

Afterwards we analyzed a constructed model of a conceptual smart meter system. The result of this analysis was a number of attack trees which discuss how each actor in the context can attempt to exploit the system. The details of these attacks were then briefly elaborated.

The following summarizes our findings and gives suggestions on how to proceed.

5.1 Privacy

As was discovered in Section 1.2.1, relatively small amounts of consumption data can reveal a lot about the consumer and his activities. Most of our actors have some interest in obtaining this information. For instance, the burglar can use information about what devices the consumer owns to decide if breaking in is worthwhile, while the electrical company wants to abuse the information in order to get a competitive edge.

In our model, all data is sent to the data hub, which means that a solution to this problem necessarily needs to include the handling of what data is sent from the smart meter to the data hub.

The granularity of the data decides how much can be learned when analyzing the data. Investigating how to protect the data by other means than granularity may be a necessary path in order to solve this problem.

5.2 Protocols

The model presented in Section 1.3 includes a lot of communication between a lot of actors. If this communication is compromised, an intruder could get access to functionality on the smart meter. A burglar could watch the security camera feed and turn off the security alarm, while a terrorist organization could turn off the power in a multitude of facilities.

In order to secure the communication contained in the smart meter architecture needs to be designed with secure protocols that fit the limited resources of a smart meter while still providing security needed to protect the data being sent through the architecture.

5.3 Data Manipulation

The integrity of the data sent between the smart meter and the data stored on the smart meter is important for the whole system to be effective. If the integrity of data cannot be relied on, the whole system loses its purpose. The consumer and the electrical company may possibly have an interest in attacking each other by manipulating the consumption data – either the stored data on, or the data sent from, the smart meter.

This is a broad problem and solutions may include designing architectures that prevent manipulating the data, protocols that ensure data has not been tampered with or firmware that is able to detect attempts at manipulation.

Bibliography

[1] R. Anderson and S. Fuloria. "Who Controls the off Switch?" In: *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on.* 2010, pp. 96–101. DOI: 10.1109/SMARTGRID.2010.5622026.

[2] Ross Anderson and Shailendra Fuloria. "On the Security Economics of Electricity Metering." In: *Workshop on the Economics of Information Security (WEIS).* 2010.

[3] Ross Anderson and Shailendra Fuloria. *Smart meter security: a survey.* https://www.cl.cam.ac.uk/~rja14/Papers/JSAC-draft.pdf. [Online; accessed 20-10-2015]. 2011.

[4] Ross J. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems.* 2nd ed. Wiley Publishing, 2008. ISBN: 9780470068526.

[5] Matt Bishop and Daniel V Klein. "Improving system security via proactive password checking". In: *Computers & Security* 14.3 (1995), pp. 233–249.

[6] David FC Brewer and Michael J Nash. "The chinese wall security policy". In: *Security and Privacy, 1989. Proceedings., 1989 IEEE Symposium on.* IEEE. 1989, pp. 206–214.

[7] David D. Clark and David R. Wilson. "A Comparison of Commercial and Military Computer Security Policies". In: *IEEE Symposium on Security and Privacy* (1987), p. 184. ISSN: 1540-7993. DOI: http://doi.ieeecomputersociety.org/10.1109/SP.1987.10001.

[8] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. "Password strength: An empirical analysis". In: *INFOCOM, 2010 Proceedings IEEE.* IEEE. 2010, pp. 1–9.

[9] TED: The energy detective. *TED installation guide.* https://www.theenergydetective.com/downloads/QuickStartInstallation%20v110711.pdf. Online; Accessed: 16-11-2015.

[10] The Energy Detective. *The Energy Detective: TED Energy Monitor.* http://www.theenergydetective.com/. Online; Accessed: 27-11-2015.

[11] Rachna Dhamija, J Doug Tygar, and Marti Hearst. “Why phishing works”. In: *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM. 2006, pp. 581–590.

[12] European Parliament Directorate General For Internal Policies Policy Department A: Economic and Scientific Policy. *Effect of smart metering on electricity prices*. `http://www.europarl.europa.eu/document/activities/cont/201202/20120223ATT39186/20120223ATT39186EN.pdf`. Online; Accessed: 25-11-2015.

[13] Efergy. *Efergy's ECM system: engage*. `https://engage.efergy.com/`. Online; Accessed: 27-11-2015.

[14] energitilsynet.dk Energitilsynet. *Marked: Energitilsynet*. `http://energitilsynet.dk/el/marked/`. Online; Accessed: 16-11-2015.

[15] energy.gov Department of Energy. *Tips: Smart Appliances*. `http://www.energy.gov/energysaver/tips-smart-appliances`. Online; Accessed: 25-11-2015.

[16] US Department of Energy. *What is the Smart Grid?* `https://www.smartgrid.gov/the_smart_grid/index.html`. [Online; accessed 20-10-2015].

[17] Niels Ferguson, Bruce Schneier, and Tadayoshi Kohno. *Cryptography engineering: design principles and practical applications*. John Wiley & Sons, 2011.

[18] Dinei Florencio and Cormac Herley. “A large-scale study of web password habits”. In: *Proceedings of the 16th international conference on World Wide Web*. ACM. 2007, pp. 657–666.

[19] James C Foster, Vitaly Osipov, and Nish Bhalla. *Buffer Overflow Attacks*. Syngress Publishing, 2005.

[20] Michael A. Harrison, Walter L. Ruzzo, and Jeffrey D. Ullman. “Protection in Operating Systems”. In: *Communications of the ACM* 19.8 (1976), p. 11.

[21] Paul Kocher, Joshua Jaffe, and Benjamin Jun. “Differential power analysis”. In: *Advances in Cryptology—CRYPTO'99*. Springer. 1999, pp. 388–397.

[22] Leonard J LaPadula and D Elliot Bell. *Secure computer systems: A mathematical model*. Tech. rep. Technical Report 2547, 1996.

[23] Johnny Long. *No tech hacking: A guide to social engineering, dumpster diving, and shoulder surfing*. Syngress, 2011.

[24] Simon Marechal. “Advances in password cracking”. In: *Journal in computer virology* 4.1 (2008), pp. 73–81.

[25] Andrés Molina-Markham et al. “Private Memoirs of a Smart Meter”. In: *Proceedings of the 2Nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*. BuildSys '10. Zurich, Switzerland: ACM, 2010, pp. 61–66. ISBN: 978-1-4503-0458-0. DOI: `10.1145/1878431.1878446`. URL: `http://doi.acm.org/10.1145/1878431.1878446`.

[26] Open Energy Monitor. *Open Energy Monitor: The Open Energy Monitor.* `http://openenergymonitor.org/emon/`. Online; Accessed: 27-11-2015.

[27] Andrew C Myers. "Mostly-static decentralized information flow control". PhD thesis. Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 1999.

[28] Andrew C. Myers and Barbara Liskov. "A Decentralized Model for Information Flow Control". In: *SIGOPS Oper. Syst. Rev.* 31.5 (Oct. 1997), pp. 129–142. ISSN: 0163-5980. DOI: `10.1145/269005.266669`. URL: `http://doi.acm.org/10.1145/269005.266669`.

[29] The European Parliament and the council of the European Union. *Directive 2009/72/EC of the European Parliament and of the council – of 13 July 2009.* `https://www.energy-community.org/pls/portal/docs/1164180.PDF`. [Online; accessed 20-10-2015]. 2009.

[30] Oliver Parson et al. "Non-intrusive load monitoring using prior models of general appliance types". In: (2012).

[31] M. L. Pedersen and M. H. Sørensen. *The Timed Decentralized Label Model.* Tech. rep. Selma Lagerlöfs Vej 300, 9220 Aalborg Øst, Denmark: Department of Computer Science, Aalborg University, 2015.

[32] retsinformation.dk. *Lov om ændring af lov om elforsyning, lov om naturgasforsyning og lov om Energinet.dk.* `https://www.retsinformation.dk/forms/r0710.aspx?id=142359`. [Online; accessed 20-10-2015].

[33] Olatunji Ruwase and Monica S Lam. "A Practical Dynamic Buffer Overflow Detector." In: *NDSS*. 2004.

[34] Bruce Schneier. *Attack Trees - Modeling security threats.* `https://www.schneier.com/paper-attacktrees-ddj-ft.html`. Online; Accessed: 30-11-2015.

[35] Seth Schoen. "Detecting packet injection: A guide to observing packet spoofing by ISPs". In: *Electronic Frontier Foundation whitepaper* (2007).