U N I V E R S I T Y   OF   T A R T U
Faculty of Mathematics and Computer Science
Institute of Computer Science

Riivo Talviste

Deploying secure multiparty computation for joint data analysis — a case study

Master's Thesis (30 ECTS)

Supervisor: Dan Bogdanov, MSc

Author:	"....."	May	2011
Supervisor:	"....."	May	2011

Allowed to defence

Professor:	"....."	May	2011

TARTU 2011

Contents

1 Introduction

Think of a consortium of companies working in the same field. Each of them has its own customer base and thus its own view of a market segment. However, to make better business decisions, these companies would also like to have a good overview of the whole market in general. For example, let us take the food market with several supermarket chains operating on that market. Each one of them has their own customer loyalty program that they use to gather data about the customers' shopping habits. They use that data to perform various shopping cart analyses (e.g. frequent itemset mining). However, this way each company can only analyze and predict the habits of its own customer base and they do not have an overview of the whole food market.

Having a good overview of the whole market could be easily accomplished by collecting data from all of those companies and computing statistics from the whole. However, as data is part of their trade secret, companies are usually reluctant to disclose their data so it could be seen and analyzed by others. Furthermore, if this data also contains customers' sensitive information, several legal acts apply that prohibit the companies from disclosing that information [2, 1, 12].

Privacy-preserving data mining and secure computation methods are viable solutions that enable this kind on data analysis without compromising the confidentiality of the data. There are several secure multiparty computation (MPC) frameworks that could be used in this scenario (VIFF [26], SEPIA [11], Sharemind [6], JSMC [8]).

In this thesis we analyze the aspects of deploying MPC applications to solve real life problems. Our work relies on a case study of developing an application for one consortium, namely the Estonian Association of Information Technology and Telecommunications (ITL)[1].

1.1 Outline

In Section 2 we introduce the concept of secure multiparty computation and give a brief overview of the Sharemind framework that we are going to use in our case study.

Section 3 describes the initial problem where a consortium of ICT companies would like to calculate benchmarking results based on their economic indicators. We point out some security-related shortcomings of the initial solution based on anonymization and propose a new solution with stronger privacy guarantees achieved by using secure MPC.

Section 4 presents a general architecture of how to build privacy-preserving

[1]`http://www.itl.ee`

applications using the Sharemind framework with data submission from the web. All of the main components of the architecture are described in detail.

Section 5 is dedicated to the JavaScript library that is used in the web based data submission application. We elaborate on how some of the main challenges like secret sharing the inserted data and sending the shares to the data miners were overcome. Finally, there is a simple tutorial on how to use the JavaScript library in a web based data submission form.

Section 6 describes how the idea from Section 3 and the architecture introduced in Section 4 were used to solve the problem that ITL had. We elaborate on how the Sharemind data miners were chosen and where all of the components are deployed. We also walk the reader through the user story of the ITL application.

Finally, Section 7 gives a timeline for the data collection and report generation processes in the ITL project. We also give a brief summary about the user feedback.

1.2 Author's contribution

In this section we list the author's original contributions to this thesis. Since the beginning of 2010 the author has participated in a Software Technology and Applications Competence Center (STACC) project. In this project we developed the first prototype of an online privacy-preserving questionnaire application using the Sharemind framework. The partner organizations in this project were Cybernetica, Swedbank and Quretec. The general architecture used in the STACC project as well as the architecture described in the Section 4 of this thesis were inspired by the work in the author's bachelor thesis written in 2009 [23]. One of the main contribution of the author in the STACC project is writing the JavaScript library that is described in Section 5.

While building the solution for the ITL, the main contributions of the author include: developing the web-based data submission form and integrating it with the ITL web page; writing the necessary proxy applications, analysis applications and SecreC scripts described in Sections 4 and 6; deploying the Sharemind system; and composing both the user manuals and deployment documentation mentioned in Section 6.5.

One of the main challenges of this work was overcoming the shortcomings that web-based applications have compared to the the desktop applications. Specifically, we had to bypass the Same Origin Policy to connect to different servers from the web application and find a way to generate cryptographically secure random numbers in the web browser. Also, the solution that we developed for the ITL is a distributed application running on three servers with many separate components on each server. Thus, we tried to make deploying and maintaining this application easier on different servers.

2 Preliminaries

2.1 Privacy-preserving data mining

Collecting and analyzing large data sets is often restricted by confidentiality requirements to avoid anyone's privacy from being breached. However, often we do not collect data for the sake of having the individual values, but rather for using various data mining algorithms to generate models that characterize the whole data set in general.

Techniques like anonymization and pseudonymization can help protect individual values [17], but they also introduce a trade-off between data quality and the privacy level. Furthermore, there are numerous incidents where identities are derived from non-identifying attributes [4, 10, 18].

Secure multiparty computation (MPC) is a technique for evaluating a function with multiple peers so that each of them learns the output value but not each other's inputs. There are various ways for implementing secure MPC with different number of peers and security guarantees. In this work, we concentrate on systems based on secret sharing (also called share computing systems).

Share computing systems use the concept of secret sharing introduced by Blakley [5] and Shamir [21]. In secret sharing, a secret value s is split into any number of shares s_1, s_2, $\ldots$, s_n that are distributed among the peers. Depending on the type of scheme used, the original value can be reconstructed only by knowing all or a predefined number (threshold t) of these shares. Any group of t or more peers can combine their shares to reconstruct the original value. However, the result of combining less than t shares provides no information about the value they represent.

Secure multiparty computation protocols can be used to process secret shared data. These protocols take secret shared values as inputs and output a secret shared result that can be used in further computations. For example, let us have values u and v that are both secret shared and distributed among all the peers so that each peer $\mathcal{P}_i$ gets the shares u_i and v_i. To evaluate $w = u \oplus v$, the peers engage in a share computing protocol and output w in a shared form (peer $\mathcal{P}_i$ holds w_i). During the computation, no computing party is able to recover the original values u or v.

Multiparty computation protocols can be secure in either passive or active corruption models. In the passive model, an adversary can read all the information available to the corrupted peer, but it cannot modify it. In this case, the corrupted peer still follows the predefined protocol, but it tries to deduce the original data values based on the information available to that peer. This is also known as *honest-but-curious* model. In the active model, an adversary has full control over the corrupted peer. For more properties of secure MPC protocols, see [13].

2.2 Sharemind

Sharemind [6] is a distributed virtual machine for performing privacy-preserving computations. The Sharemind framework can perform various operations on 32-bit integers and vectors of 32-bit integers.

The Sharemind framework allows the developer to write algorithms where public and private data are separated. The Sharemind virtual machine guarantees that private data is not leaked while such an algorithm is evaluated.

The Sharemind system uses three peers to hold the shares of secret values. In Sharemind terminology, these peers are *miners*. The miners are connected with each other by the network and use secure MPC protocols to evaluate a function on the secret shared data. The Sharemind computation protocols are provably secure in the honest-but-curious model with no more than one corrupted party.

Secret sharing of private data is performed at the source and each share is sent to a different miner over a secure channel. This guarantees that no-one except the data source will know the original value. Sharemind uses additive secret sharing scheme in the ring $\mathbb{Z}_{2^{32}}$. Suppose, that we have a secret value s that we want to share. Let us also assume that we have a function `random()` that gives us uniformly distributed random values from the ring $\mathbb{Z}_{2^{32}}$. Additive secret sharing works like this:

```
$s_1 \leftarrow random()$
$s_2 \leftarrow random()$
$s_3 \leftarrow s - s_1 - s_2$
```

All calculations are done modulo 2^{32}, so s_1, s_2 and s_3 are also in the ring $\mathbb{Z}_{2^{32}}$.

Figure 1 shows the most common deployment scenario for the Sharemind framework. Assume that we have several parties who all have their private data and are interested in the statistical data analysis results involving data from other parties. They choose three representatives among themselves who are motivated to participate in this collective endeavor but at the same time will not collude with each other. This restriction is necessary to retain the privacy guarantees of the Sharemind framework. These three parties will host the miners. All participating parties secret share their data and distribute the shares among the three miners.

The Sharemind framework provides the developers with a controller library. This library is used by both data sources for inserting the data to the system as well as analysis applications that order miners to run data analysis algorithms which are evaluated by using secure MPC protocols. The analysis applications only receive the final results of the computations. Therefore, after secret sharing the original data at the source, no party will learn individual values inserted by those sources, unless they are explicitly made public in the analysis algorithms.

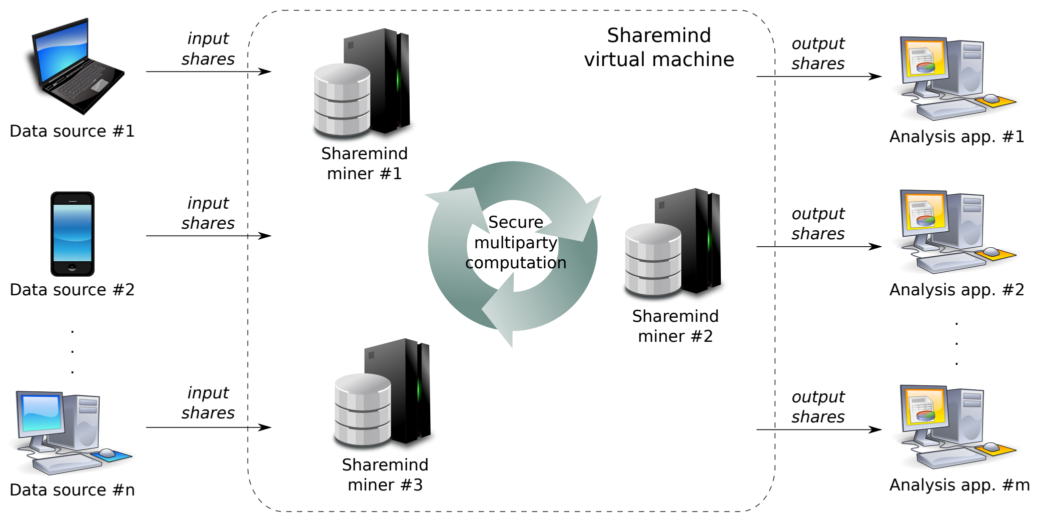


Figure 1: Deployment model of the SHAREMIND virtual machine.

2.3 Sharemind application's development process

Creating applications with the SHAREMIND framework involves three main steps. Firstly, we have to find three independent parties who will host the miners. Each of those hosts has to set up a server and install SHAREMIND miner software on it. All three miners have to be configured properly (firewall rules, encryption keys, addresses and port bindings) so they can successfully communicate with each other.

Secondly, we have to develop the necessary data-mining applications that take advantage of the privacy-preserving guarantees that the SHAREMIND framework provides. SHAREMIND has a low-level assembly language that the virtual machine can execute. As implementing an algorithm in low-level language is tedious and error-prone, the framework also provides the developers with a more high-level programming language named SECREC. SECREC [16] is a high-level language with C-like syntax that is capable of separating public and private data flows. It means that the public computations are done in an ordinary manner, while private computations involving sensitive information (shares of secret values) are evaluated using secure MPC protocols. SECREC applications are compiled into SHAREMIND assembly, which is then given to each SHAREMIND miner and that can be then executed by the SHAREMIND virtual machine.

Writing privacy-preserving programs is more convenient with the SECRECIDE integrated development environment [20], which provides basic project management, syntax highlighting and debugging support.

Thirdly, we need to use the SHAREMIND controller library to build end-user applications. These applications are used to insert the data into the SHAREMIND miners, run analysis on that data and also view the results.

3 Joint analysis of financial data

3.1 Problem statement

The Estonian Association of Information Technology and Telecommunications (ITL) is a non-governmental non-profit organization whose primary focus is to unite Estonian information technology and telecommunication companies and promote their co-operation. ITL has over 60 members – Estonian enterprises that engage in the field of information and communication technology (ICT).

ICT is a fast-growing industry and thus ITL members would like to be able to compare themselves with other companies in that sector more often than once a year when the Ministry of Economic Affairs and Communications releases the annual economic reports. Making business decisions based on this annual report means making decisions based on data that is about half a year old and this is not sufficient for such a dynamic industry as ICT.

3.2 The initial solution and its security

There is an initiative within ITL proposing that the organization should collect some basic economic indicators from its members (in the beginning, only IT companies) twice a year. Based on these collected indicators, ITL can release anonymized benchmarking info for each collected indicator to its members. This enables businesses to compare themselves with the best in the ICT sector.

ITL collects these economic indicators from its members with the following frequency:

Indicator	Collected
total return	semi-annually
number of employees	semi-annually
percentage of export	semi-annually
added value	semi-annually
labour costs	annually
training costs	annually
profit	annually

All these seven indicators will be released to the public half a year later anyway, as they are part of the annual economic report released by the ministry. However, if ITL collects these indicators by itself, its members would be able to compare themselves with each other based on more up-to-date information.

Before publishing the results, each indicator is sorted independently to reduce the risk of identifying some companies by just looking at a set of economic indicators. For example, combining total return, number of employees and profit, it could be easy to identify some IT companies. However, when sorting by each indicator independently, a company that is first when sorted by one indicator might not be first when sorted by another indicator. Let's assume that three companies add the following records into the database:

```
(company_id=1, field1=111, field2=1234, field3=42314)
(company_id=2, field1=222, field2=2431, field3=12345)
(company_id=3, field1=212, field2=3132, field3=54321)
```

By removing the identifying information (company ID) and sorting each field separately, we get the following table where links between the values of a single row have been removed:

field1	field2	field3
222	3132	54321
212	2431	42314
111	1234	12345

The data flow and visibility to different parties is shown on Figure 2.

Sorting the collected data by each indicator separately gives us a slightly stronger privacy guarantees than just stripping away the identifying information (i.e. company name and ID in ITL database). However, all of the collected data is accessible by ITL board, which consists of the leaders of competing ICT companies. Therefore, ITL member companies might be reluctant to participate and give away their sensitive economic information.

3.3 Improved solution

To address this problem, we use the Sharemind framework to collect and analyze the economic information (see Figure 3). That way, all of the collected private information is secret shared at the source and distributed among three Sharemind miners. Using the Sharemind framework gives us the additional benefit that no single party has access to the original data values and this lowers the risk of anyone misusing the collected information. Also, we have a much lower threat of insider attacks and unintentional disclosures (i.e. compromise of economic data by a leaked backup).

After the data has been collected from all of the members, three data miners engage in secure MPC protocols and sort all the collected economic indicators independently. These sorted indicators are then published as a spreadsheet and made accessible to the board members of the ITL. The board will then either give these spreadsheets directly to all of the members or first calculate some aggregate values and/or charts and give this edited report to the members. Making the spreadsheet with the sorted values initially only available to the board members is a necessary precaution to ensure that there is no data leakage. As ITL board members are elected, they are also trusted by the rest of the ITL members. However, even the board members will not see any identifying information, as this is removed while sorting the collected values. The latter is the main advantage of the described solution using the SHAREMIND framework over the initial solution using only anonymization techniques.

Note, that there is an opportunity for further improving privacy of the collected data. Currently, ITL requires the original values that are independently sorted by each economic indicator, as it is not sure which further analyses need to be done on the results. After the first data collection period, when ITL board has agreed on the analyses to be done, we can implement these analyses with the secure MPC protocols in the SHAREMIND framework. In this case, we would not have to disclose the anonymized data vectors with the original values anymore.

The described solution was proposed to ITL board as an alternative to their initial solution, as it has stronger privacy guarantees. The board accepted the proposal and we started to develop the necessary applications. In the following sections we describe the architecture, its components and their deployment in more detail.

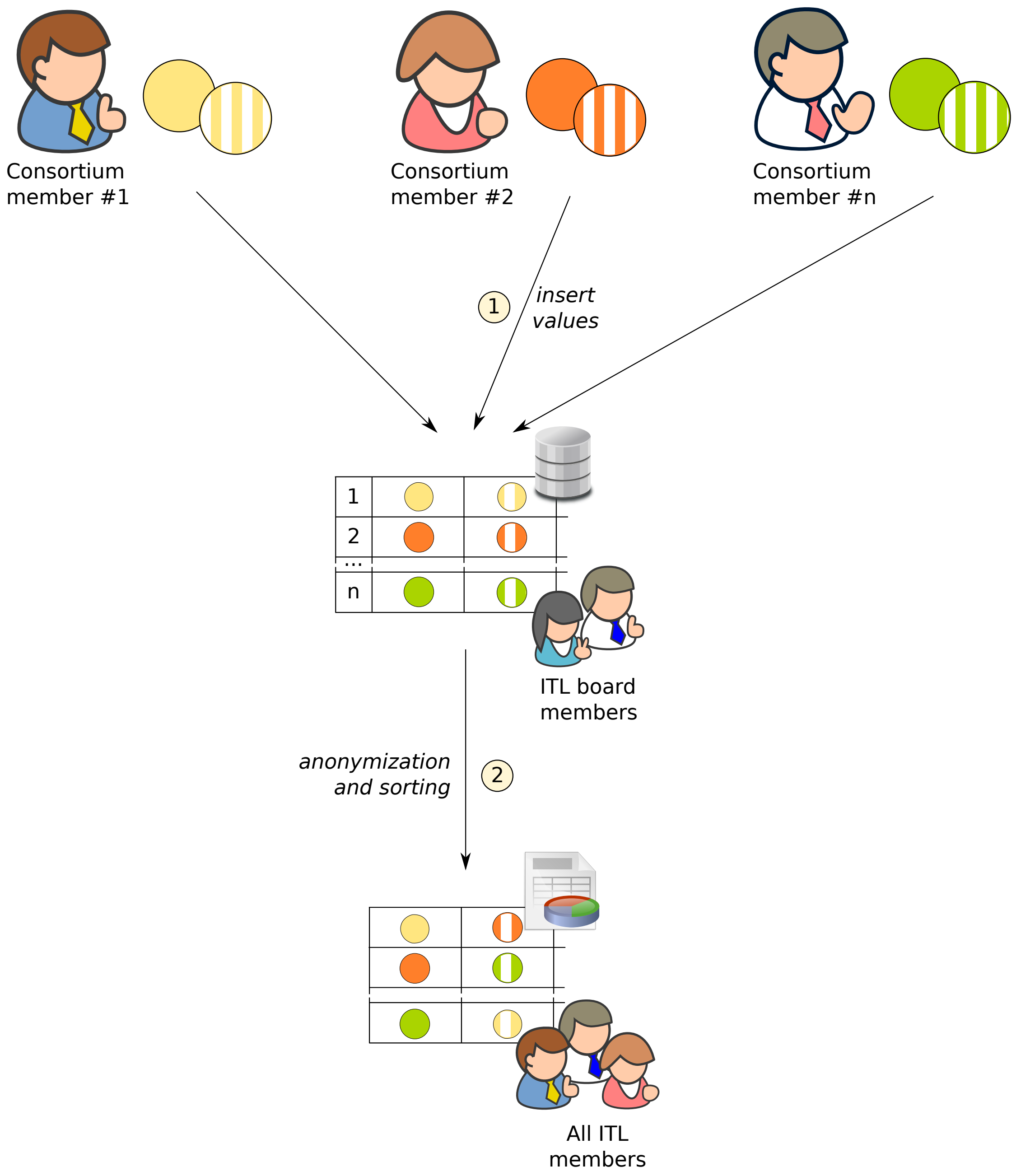


Figure 2: Data flow and visibility in the initial proposed solution.

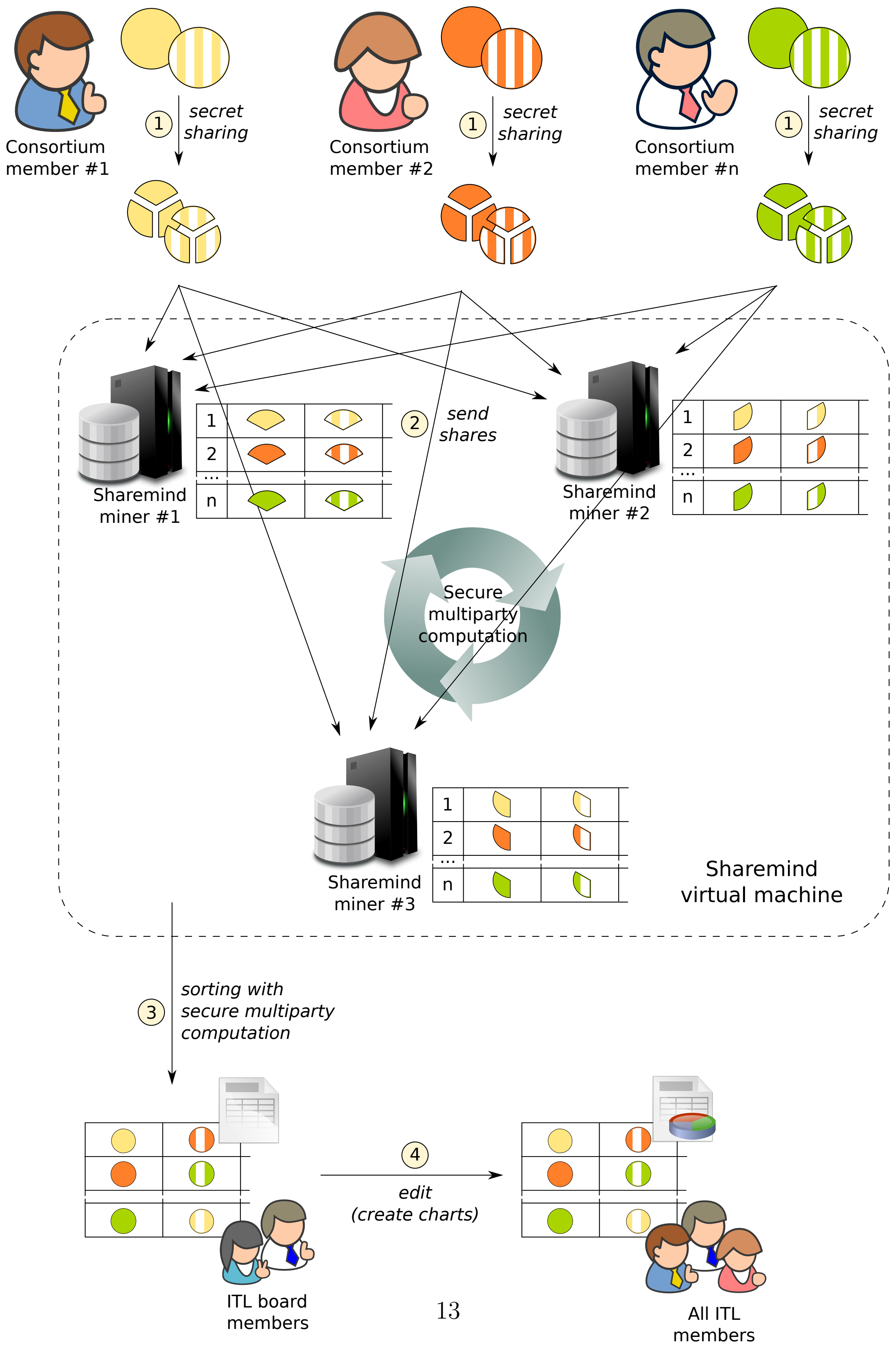


Figure 3: Data flow and visibility in the improved solution using the SHAREMIND framework.

4 Application architecture

4.1 Overview

This section describes a possible architecture for solving the problem introduced in Section 1. We have a consortium of companies where each member possesses its own data but is also interested in aggregate results involving data from other members. However, the members cannot share their data with each other either because they are reluctant or prohibited to do so.

In our earlier work [23, 24] we presented a web-based architecture where the privacy of the data is preserved throughout the whole process — from data source to analysis. Since then, we have improved the solution by enhancing its usability, robustness and security guarantees (for details, see Section 5). The general deployment scheme for this solution is shown on Figure 4.

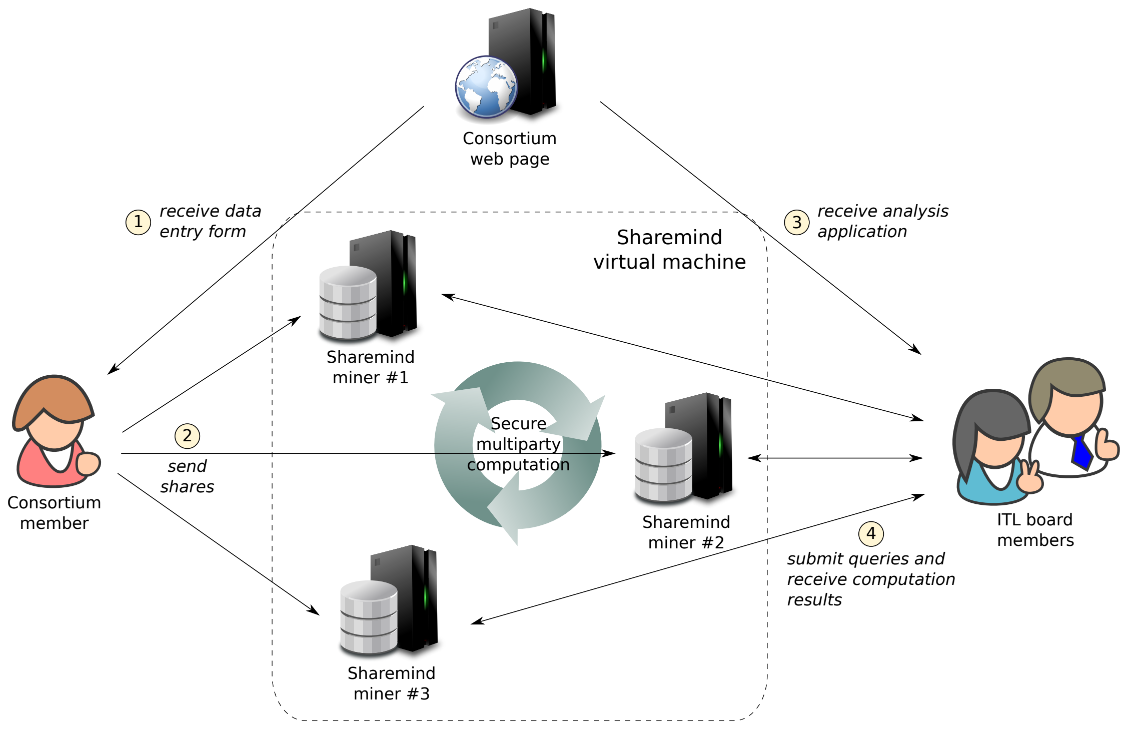

Consortium web page
1 receive data entry form
3 receive analysis application
Sharemind virtual machine
Sharemind miner #1
Secure multiparty computation
Sharemind miner #2
Sharemind miner #3
Consortium member
2 send shares
ITL board members
4 submit queries and receive computation results

Figure 4: Deployment model for the architecture using the Sharemind framework.

The consortium web page is accessible by both its members and its board members. Firstly, the consortium members receive the data entry form from the web page and and fill it. Upon submitting this form, all of the inserted values are secret shared and the shares distributed among the three Sharemind miners.

The miners are usually hosted by three members of the consortium. However, they must be motivated not to collude with each other.

Secondly, the consortium board members can download an analysis application from the consortium web and use it to make queries to the SHAREMIND virtual machine. These queries start the secure multiparty computation process. Upon finishing the computations, the miners reply with the corresponding aggregated results.

In real deployment scenario, each SHAREMIND miner shown on Figure 4 is actually a (virtual) server consisting of the following parts (see Figure 5):

- A web server with miner's web front end that receives shares from the submission form and stores them in a buffer database.
- A proxy application that relays shares from a buffer database to the miner's internal database. The proxy applications are used for increased system robustness and security.
- SHAREMIND miner instance with its internal database.

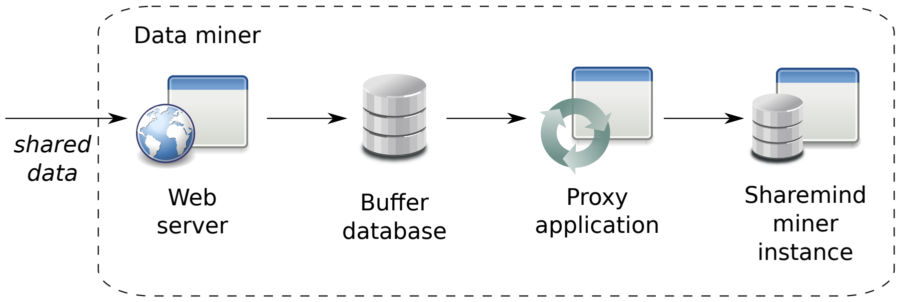


Figure 5: Components of a data miner: web server, buffer database, proxy application and a SHAREMIND miner instance.

In the following we will describe all of these components and their roles in more detail.

4.2 Miner's web interface

SHAREMIND miners only work with data from their own databases and it is up to the application developer to guarantee that the right data is in each miner's database. To simplify this process, the SHAREMIND framework provides a C++ controller library that is used to communicate with the miners. Among other things, it allows to insert data into each miner's database. However, we would like to insert data into the miner's databases from the web browser and the current version of the Sharemind controller library does not have a web interface. Thus, we have to take extra steps to receive data from the web and get it into the miner's internal database.

For receiving the shares from the online data entry application (submission form), we have coupled each Sharemind miner with a web server and a simple web application. This application receives the shares and a randomly generated 32-bit session identifier, saves them in a buffer database and replies to the submission form with either a success or an error message. This acknowledgment message is required, as the Sharemind miners are distributed across the internet and the online submission form has to know if all of the shares were successfully saved to avoid data loss. Section 5 gives a more detailed overview of the submission form and its communication with the miner's web interface.

4.3 Miner proxy application

Each of the miners is also coupled with a proxy application that loads all the shares from the corresponding buffer database and uses the controller library to insert them into the Sharemind miner's internal database where they can be used for private computations. However, the Sharemind virtual machine requires that all three miners have shares from the same data rows and in the same order. It must not happen that shares from a single data row are present in one or two miners but not in all three. This restriction is necessary as the data rows in the miners' databases do not have primary keys or any other identifying information.

To keep the miner's internal database consistent, the proxy application contacts the other two proxies running at the same time after it has loaded all the data rows from the buffer database. Together the three proxy applications find the intersection of their data rows and sort the result by session identifiers that each data row has. After that each proxy saves the result to the corresponding miner's internal database. During this synchronization process, no secret shared values are leaked as each proxy application is tied to one miner and does not communicate with the other two miners directly. Furthermore, calculating the intersection of the data rows is done using only the session identifiers and no shares of secret values are sent to each other.

This proxy application can be either run periodically or only once after the data collection period has ended and before the secure multiparty computations are performed.

Using buffer databases and proxy applications also makes the data collection process more robust, as we do not have to use any locking or commiting protocols to ensure that all miners have the same order of data rows. It also allows us to clearly separate the data collection and analysis processes so that the Sharemind miners do not even have to run (listen for connections) while the data is being collected.

In the future, we still plan to embed a simple web server into the SHAREMIND miners, so they would be able receive data directly using the HTTPS protocol. This reduces the architectural complexity of the solution and simplifies the deployment. Also, it would enable us to develop privacy-preserving applications working in nearly real-time.

4.4 Miners and data mining algorithms

SHAREMIND miners are server applications that are listening on a specific port to receive commands from the controller applications and messages from each other. The analysis application is a controller application built with the SHAREMIND controller library. It orders the miners to start evaluating a predefined algorithm using secure MPC protocols and then receives the final results. This analysis application is usually hosted by the party that is interested in the final analysis results. All the connections between miners and controller applications, as well as between the miners themselves, are encrypted.

The data analysis algorithms are written in SECREC language, compiled into SHAREMIND assembly and given to each miner so it can be executed by the SHAREMIND virtual machine.

In our scenario, the data analysis is performed right after the data collection period has ended. First, the proxy applications synchronize the data sets of the miners. After that, the analysis application starts the secure multiparty computation process. Upon receiving the final computation results, the analysis application saves them to another database in the consortium network. Using this database makes it easy to present the results in different formats, e.g. as a web page, PDF report or a spreadsheet for further analysis.

Alternatively, the proxy and analysis applications can be run periodically throughout the data collection period. This way the computation results are constantly updated to reflect new data records. However, running the proxy and analysis applications too often introduces a new security risk. A malicious party who has access to the analysis application's output, might compare the consecutive computation results and deduce some individual values from it. For example, if we use the analysis application to calculate the average value and only one number is added between two consecutive computations, it is easy to find out the added number. Therefore, depending on the domain requirements, we should pick the longest possible interval to update the computation results. This lowers the risk of an adversary successfully using the aforementioned attack.

4.5 Presenting the results

The final computation results stored by the analysis applications are used to create reports for analysts. A report could be either a spreadsheet, a PDF document or a web page and include tables, charts or other visual elements created from the secure multiparty computation results. However, since the details of its form and contents are specific for each application, we will bring an example of the report generated for the ITL in Section 6.4.

5 Web-based controller library

5.1 Introduction

As discussed in Section 4.2, the Sharemind miners do not have a web interface. However, we would like to conduct web-based surveys and use other applications where data is collected from web applications. Thus, we have to provide the Sharemind miners with a web interface ourselves. We chose to couple each Sharemind miner with a simple web application that receives shares from the data source (i.e. an online data submission form) and relays them to the miner itself.

Since we are working with sensitive data, we have to be extra careful when sending private data across the web. The protocols used by the Sharemind virtual machine are proved to leak no information. However, we also have to make sure that we do not compromise the privacy of the shared data between secret sharing and saving it to the miner's database. No one except the data owner should be able to see the original input values. To accomplish this, the data has to be secret shared at the source and each share sent to a miner so that the communication cannot be eavesdropped.

In our proposed solution the data owner uses an online data entry form to submit his or her sensitive information (as seen on Figure 4 in Section 4). The form is a HTML form with a JavaScript library that handles secret sharing for the inserted data and sends the shares to the miners.

A similar system for collecting data for secure multiparty computation was developed in Denmark in 2008 [7]. They used a Java applet for entering data using a web application. However, we have chosen JavaScript as our development platform, as it is available in most web browsers by default and we do not depend on any browser plug-ins (e.g. Adobe® Flash®, Microsoft Silverlight®) or other software installed in the end user's computer (e.g. Java). Moreover, using only HTML and JavaScript makes our data entry application also usable from mobile devices without any additional effort.

In this section we explain how the data submission form and the JavaScript library work.

5.2 Connecting to the miners

One of the tasks of the JavaScript library is to communicate with each miner's web interface. However, all modern web browsers enforce the Same Origin Policy for JavaScript applications. This policy restricts JavaScript applications from making any connections outside the web page origin domain (i.e. the domain that the web page was loaded from). So the Same Origin Policy only allows to make connections to web pages that have the same domain name (subdomains are also allowed), port and protocol as the origin domain.

This is a problem, as our JavaScript library needs to connect to three different miner web servers to securely transmit shares. Since the SHAREMIND deployment model requires that the three miners are independent, it is most probable that all of the miners are deployed in different domains, so each miner host has exclusive control over its miner. This means that making simple `XMLHttpRequest` queries to one of the miners from our JavaScript library would just fail.

Next, we will describe two methods to overcome the restrictions enforced by the Same Origin Policy — the so-called *script tag workaround* and HTML5 cross-document messaging API.

5.2.1 Script tag workaround

The HTML resource tags (e.g. `<img>`, `<script>`) are not affected by the Same Origin Policy, as one may include images and scripts from any domain in his or her web page. Using the `<script>` tag to communicate with web pages from remote domains is often called *script tag workaround*.

To use this workaround, we have to create a new `<script>` HTML node and set its `src` attribute to the web page we want to send data to. The data itself should be encoded as query parameters. The GET request to the specified URL is made automatically right after the created node is inserted to the Document Object Model (DOM) of the active web page.

One of the restrictions of this approach is that the dynamically loaded page has to be a valid script file, as it gets executed when the loading is complete. So to receive more complex data structures, it is wise to encode the data in JavaScript Object Notation (JSON) [14] format.

Since the `<script>` tag is dynamically added to the DOM and the page is loaded in the background, it is also difficult to know when exactly it has finished loading. One of the solutions is to use *JSON with padding* (JSONP). The idea behind this approach is that the loaded data (JSON-encoded body of the loaded script file) is encapsulated in a JavaScript function call. For example, suppose that we have a predefined JavaScript function `dataHandler` in our web page. The body of the loaded script file should look similar to:

```
dataHandler(<JSON-encoded data>);
```

This way the function `dataHandler` is executed automatically with received data as argument when the dynamically loaded script is loaded.

In our solution we use the Dojo Toolkit [15] implementation of the script tag workaround. The Dojo Toolkit is a general purpose JavaScript framework that speeds up the development process of a web service and it supports all major web browsers. Dojo's implementation of the script tag workaround also supports JSON with padding.

As we are dealing with sensitive data, we are using the script tag workaround over a secure communication channel, i.e. we are using the HTTPS protocol. In this case, we have to keep in mind that the SSL certificate of the miner web server has to be already trusted by the user's web browser. Normally, when a user goes to a web page who's certificate is not trusted by the user, he or she is presented with a choice to either accept the untrusted certificate or leave the page. However, since the `script` tag is inserted to the HTML DOM dynamically by JavaScript, there is no way for the browser to ask user's permission to use an untrusted certificate for that connection. In the latter case, the connection fails silently and a timeout handler (if any) is triggered.

5.2.2 HTML5 cross-document messaging API

With the new HTML5 standard emerging, there is no need for the script tag workaround anymore. HTML5 introduces a new cross-document messaging API [27] that allows documents to communicate with each other regardless of their domain, but still in a controlled way to prevent any cross-site scripting attacks. However, since HTML5 cross-document messaging is only supported by the latest versions of web browsers[2], we would like to retain also the support for the script tag workaround. Thus, the JavaScript library we have developed first checks if the function `window.postMessage()` is defined. If it is, the library uses the enhanced capabilities of HTML5, otherwise, it falls back to using the script tag workaround.

To use the message-passing API, a web page has to have an `iframe`, an inline frame element that can load another document in it. The main page and each `iframe` can load pages from different domains and they have separate contexts. However, they can communicate with each other using the HTML5 cross-document messaging API.

To retain the compatibility with the script tag workaround and avoid writing the same code twice, we use simple proxy web pages for message-passing. These proxy pages are located in the same remote domains as the miner web interface scripts that need to be accessed. Our data submission form page creates three hidden `iframe`-s, each of which loads a proxy page from a corresponding miner server. All of the messages between the form and miners' servers are first passed to the proxy pages via the message-passing interface and then forwarded to the correct endpoint with a simple `XMLHttpRequest` query made by a proxy page. The response messages are also sent back to the initial form with the help of the message-passing interface. The messages themselves are the same JSON-encoded

[2]Web browsers supporting HTML5 cross-document messaging API: Internet Explorer 8+, Firefox 3.5+, Safari 4+, Chrome 8+, Opera 10.5+. Data from `http://caniuse.com`

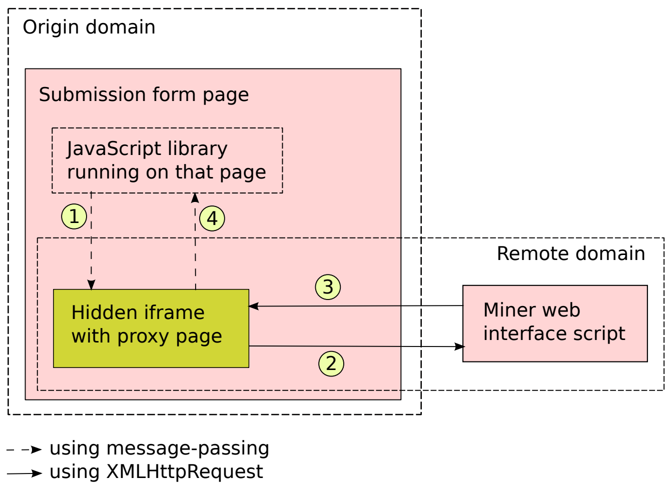

Origin domain
Submission form page
JavaScript library running on that page
1
4
Remote domain
3
Hidden iframe with proxy page
Miner web interface script
2
using message-passing
using XMLHttpRequest

Figure 6: Communication flow using HTML5 cross-document messaging. All messages use JSON-encoded data.

messages used with the script tag workaround. Figure 6 depicts the communication flow when using message-passing capabilities.

As with using the script tag workaround, we load the main page and also the proxy pages in `iframe`-s using the HTTPS protocol. This means that the `XMLHttpRequest` queries made by the proxy pages also use secure communication channels. Again, the SSL certificates of those pages must be already trusted by the user's web browser, as the `iframe`-s are hidden and the user has no way to confirm a security exception in this case. The message-passing communication between the main page and `iframe`-s uses domain names to make sure that the messages originate from the correct source. This communication is not cryptographically secured as it happens in the user's web browser where no one can eavesdrop it.

5.3 Random number generation

In Section 2.2 we showed how to implement an additive secret sharing scheme if we have a `random()` function that gives us uniformly distributed 32-bit random integers. However, when we are trying to implement a secret sharing scheme or any other randomised cryptographic primitive in a client-side web application, we run into some problems, as we need to have a real implementation of that `random()` function.

The first option is to use some web technologies that have built-in random number generators (RNG). For example, an analogue of `Math.random()` is present in most programming languages. However, it does not satisfy our needs, as the implementation of `Math.random()` in most web browsers is based on a linear congruential generator (LCG). It has been shown that pseudo-random number generators (PRNG) based on LCGs are vulnerable to attacks [9]. Instead, we have to use cryptographically secure random values i.e. values that could be used in cryptographic primitives like key generation, etc.

Modern operating systems provide pseudo-random number generators as a system service, as it is more feasible for the operating system to collect entropy from various hardware events and user input. Both `/dev/random` in Linux systems and CryptoAPI in Windows are capable of generating cryptographically secure random values.

For example, the Java programming language can be used to develop web applets, which run on client-side. The Java virtual machine also has access to the PRNG provided by the operating system. Another possibility is to use the Microsoft Silverlight platform to write interactive web applications that run in user's web browser. Silverlight has access to a subset of Microsoft .NET framework, which also includes `RNGCryptoServiceProvider` class that is able to generate cryptographically secure random data using the operating system's PRNG. However, we chose not to use Java applets or Silverlight in our project as we wanted to keep it lightweight and not dependent on any browser plug-ins.

The second option is that the user's web browser receives some random data together with the submission form from the server. This works, as the server definitely has access to the PRNG provided by its operating system. The web application running in the user's browser could then use this random data directly for secret sharing the answers or use it to seed some pseudo-random number generator for later use. However, this way the server knows all the random values used by the web application and it would not need all the shares to reconstruct the original values.

Another viable solution is to use some JavaScript cryptography library that is able to generate cryptographically secure random numbers. For example, jsCrypto [22] is able to extract entropy from the user's interaction on a web page, specifically mouse movements. A survey conducted by the developers of the jsCrypto library showed that while users are filling in an online survey, their mouse movements generate enough entropy to seed a PRNG.

While using such a library seems easier, we propose another solution. Using the Sharemind framework means that we have to have three independent data miners that will not collude with each other. Thus, we ask each Sharemind miner for 32 bytes of random data, combine it, and use the combined results for initializing the Advanced Encryption Standard (AES) block cipher in counter mode. This way we have constructed a standard PRNG [3] without any single server knowing the initial seed.

In details, when the web page with the submission form has finished loading, it uses the aforementioned script tag workaround (or HTML5 cross-document messaging, depending on the web browser's capabilities) to ask each miner for 32 bytes of random data. The received random values are then bitwise XOR-ed together and the result is split into two 16-byte parts. These 16-byte parts are used as a key and nonce respectively to initialize AES in counter (CTR) mode. We use the jsaes [19] JavaScript implementation of the AES algorithm. The resulting AES cipher text can be split into 4-byte chunks and used as 32-bit values needed for secret sharing.

5.4 Using the library

The JavaScript library described in this section (see Appendix E for the source code) can be easily used by web developers to create online data entry applications with the ability to perform secret sharing and securely distribute the shares among the miners. As we used the Dojo Toolkit implementation of the script tag workaround, we decided to write the whole library as a Dojo module. This makes using our library more convenient and the developer has all of the other Dojo functions at his or her disposal.

At first, the developer should create a web page with a simple HTML form including all the required data fields, for example:

```
<html>
  <head></head>
  <body>
    <form method="POST" action="">
      <input id="input1" type="text" />
      <input type="submit" value="Send" />
    </form>
  </body>
</html>
```

After this web page has loaded, it should load the main class of our Dojo module and make a new instance of it:

```
dojo.require('stacc.PrivateSurvey');

var hosts = [
  'https://www.host1.com',
  'https://www.host2.com',
  'https://www.host3.com'
];
```

```
var paths = [
  '/survey/',
  '/survey/',
  '/survey/'
];

var survey = new stacc.PrivateSurvey(hosts, paths);
```

Right after the new instance is created, it starts to collect randomness from the three miners. Depending on the web browser's capabilities, it uses either the script tag workaround or HTML5 cross-document messaging to do this. To get randomness, the library makes a query to a `random.php` script present on each miner's web interface. This script outputs 32 bytes of random data in a JSON-encoded message:

```
{"miner":1,"random":"1c 43 aa 30 85 5e 89 6c a8 96 77 73 7f aa 47
  db a8 52 5e a4 9d a0 d8 f1 0e b1 b9 8f fd 48 52 c4",
  "success":true,"errno":0,"error":""}
```

The library uses Dojo events to notify when it has finished doing something. For example, the library creates a `ready` event when it has finished collecting the randomness and successfully initialized the AES block cipher. The developer can create handlers for those events:

```
dojo.connect(survey, 'ready', null, function() {
  // Randomness is collected, do something useful.
  // For example, enable the 'Send' button that should be disabled
  // at the beginning, as we cannot perform secret sharing
  // without initializing PRNG first.
});
```

When the form is filled and the "Send" button is clicked, the form must not be submitted as usual, but rather a JavaScript function should be called. This function should collect all the inserted values from the form and add them to an `AnswerSet`. `AnswerSet` is a collection of key-value pairs to hold the user-inserted values. The current version of SHAREMIND supports only one data type, namely the 32-bit unsigned integer. However, in some applications, we would also like to work with negative numbers. This is also the case with the ITL application, as the economic indicator "profit" can be negative. To work with negative numbers, we treat the unsigned integers of the SHAREMIND virtual machine as 32-bit signed integers. When a negative number x is added to a `AnswerSet`, it is replaced with a unsigned integer $x' = x \mod 2^{32}$ instead (using non-negative modulus). Of course, in this case the analysis application has to take into account that values equal to or greater than 2^{31} represent negative values and we have to subtract 2^{32} to get the right value.

After adding all the user-inserted values to an `AnswerSet`, the function should invoke the `send()` function of the previously created `PrivateSurvey` instance:

```
function onSubmit() {
  dojo.require('stacc.AnswerSet');

  // Make new answer set for private values:
  var as = new stacc.AnswerSet();

  // Add the user-inserted values
  // For shortness, we omit any input validation here
  as.addValue('input1', parseInt(dojo.byId('input1')));

  // Perform secret sharing and distribute the shares:
  if (!survey.send(as)) {
    // PrivateSurvey.send() returns false if there PRNG is not yet
    // initialized. In this case we create a new 'failure' event.
    survey.failure();
  }
}
```

Upon calling the `send()` function, all the values added to the `AnswerSet` are secret shared and a JSON-encoded message including the corresponding shares is constructed for each miner:

```
{"type":"save","session":"1477273566","shares":"{
"input1_submitted":"4130807163","input1_value":"2021162429"}"}
```

The `input1_value` represents the user input value, while `input1_submitted` represents a boolean value (either 1 or 0) of whether this field was actually filled by the user or not. Since the information about which fields are filled by the user and which are not, is considered sensitive, this value is also secret shared.

This message is passed on as a query parameter to the `save_jsonp.php` script at miner's web interface. After successfully saving those into the buffer database, the script answers with a similar JSON-encoded message:

```
{"miner":1,"session":"1477273566","data":[],"shares":{
"input1_submitted":"4130807163","input1_value":"2021162429"},
"success":true,"errno":0,"error":""}
```

If the JavaScript library receives that kind of success message from each miner, it creates the `success` event. However, if at least one of the miners is unable to save the shares and return an error message, the library creates a `failure` event instead. Similarly to the `ready` event, the developer can create handlers for those events, for instance, to let the user know if the data was successfully saved or not.

6 Deployment analysis

6.1 Miner setup

To use the Sharemind framework and the architecture described in Section 4, ITL members must first choose three representatives among themselves, who will host the Sharemind miner software. Choosing the three miners among the ITL members themselves fulfills all the prerequisites, as:

1. They are motivated to host the miners, as this project would also be beneficial for themselves.

2. They are independent and will not collude with each other as they are also inserting their own data into the system and want to keep it private.

3. Also, ITL members act on the field of information technology, thus they have the necessary infrastructure and competence to host a server that runs a Sharemind miner.

For this project Cybernetica, Microlink and Zone Media were chosen to be the hosts of the miners. Both Microlink and Zone deployed a virtual machine that acts as a Sharemind miner in their domain (the corresponding servers are `itl.microlink.ee` and `itl.zone.ee` respectively). Cybernetica uses its research server (`research.cyber.ee`) as the miner. All of those servers have an installation of the Sharemind miner, the proxy application, a web server (Apache) with miner's web frontend application and an SQL database engine (MySQL) for the buffer database.

6.2 Data collection

ITL has a web page `http://www.itl.ee` that has both public pages and a member area. All the representatives of ITL member companies can log in to the member area to access the ITL event calendar, internal documents, etc. In this project, we decided to integrate the data submission form into the ITL web page member area, as it would improve usability for the representatives of the member companies. This way, ITL members can access everything related to ITL from one place and the environment is also more familiar. Moreover, it allows us to reuse the authentication mechanisms of the ITL web page without implementing them on our own. Thus, the users can access the submission form with the credentials they already have.

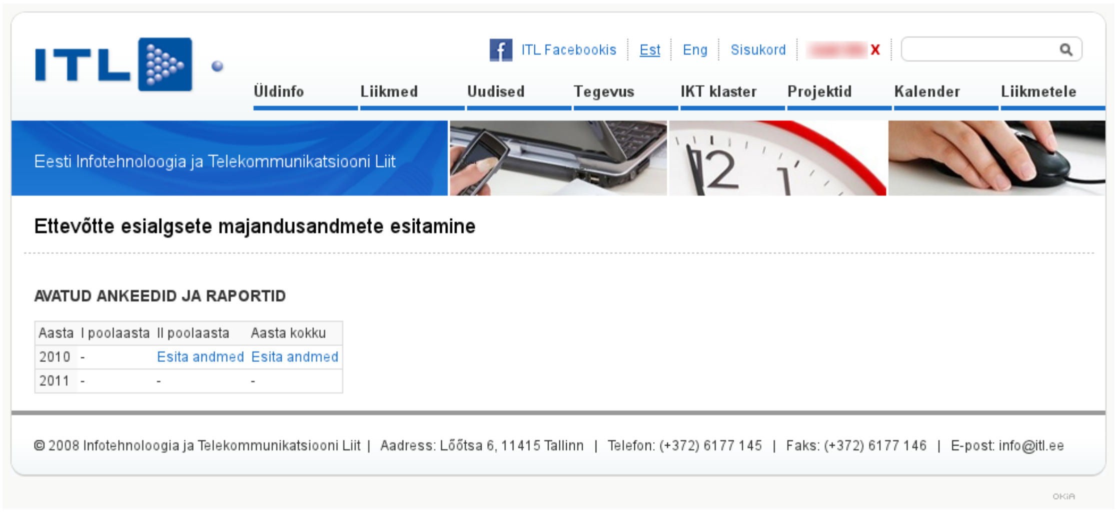
The ITL web page uses a content management system (CMS) written in PHP. By now, this CMS is a legacy software and is no longer supported by its developers.

As this CMS software is not modular, integrating the submission form required us to also make some small changes in the CMS code itself.

6.3 User story

6.3.1 Filling in the form

An ITL member can log in to the member area of the web site with his of her credentials. If the authenticated user has the rights to submit the economic indicators (i.e. the user belongs to the corresponding group), he or she can read about the goals of collecting economic indicators as well as about the security measures taken to protect the collected data. After that, the user can go to a page where he or she is presented with a list of all open economic indicator submission forms (see Figure 7). According to the initial plan, a submission form is opened for 45 days after a year has ended (from Jan 1st to Feb 14th) and for 30 days after a half-year has ended (from Jan 1st to Jan 30th and from July 1st to July 30th).



Ettevõtte esialgsete majandusandmete esitamine

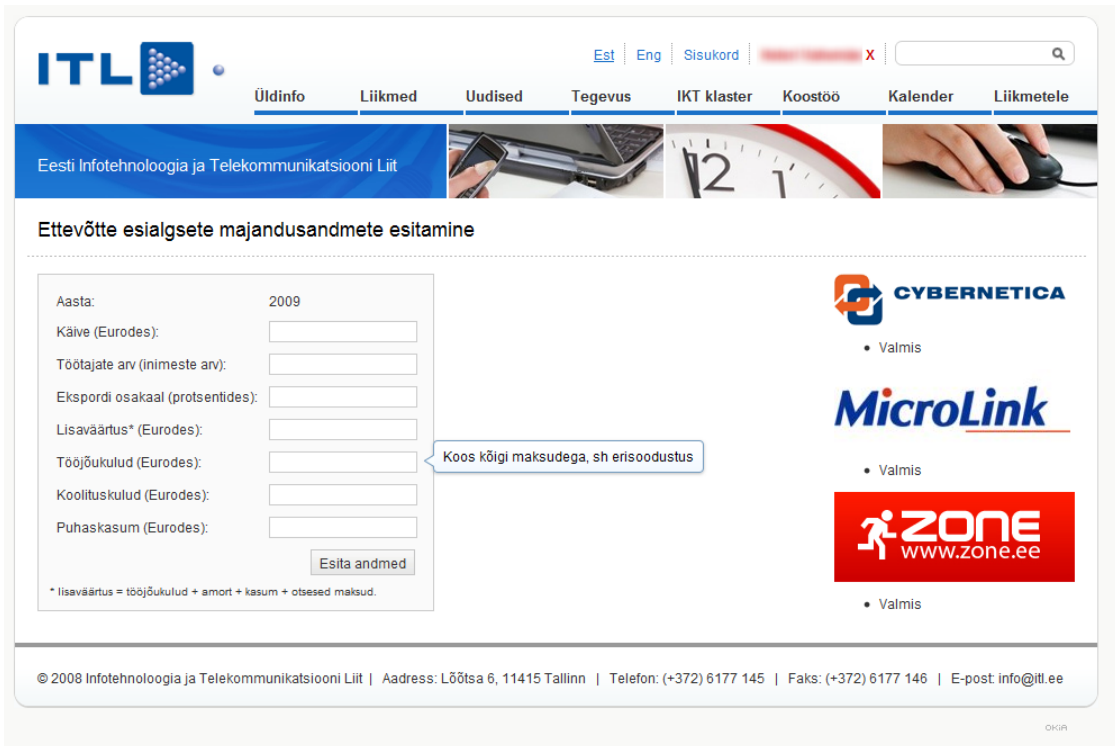
AVATUD ANKEEDID JA RAPORTID

Aasta	I poolaasta	II poolaasta	Aasta kokku
2010	-	Esita andmed	Esita andmed
2011	-	-	-

© 2008 Infotehnoloogia ja Telekommunikatsiooni Liit | Aadress: Lõõtsa 6, 11415 Tallinn | Telefon: (+372) 6177 145 | Faks: (+372) 6177 146 | E-post: info@itl.ee

Figure 7: ITL web page member area. List of all opened forms. This screenshot is taken in January so both the full year form of 2010 and the form for the second half of 2010 are open.

Clicking on the link of an opened form loads the form. The submission form itself is a simple HTML form with a text field for each of the economic indicators (see Figure 8). Next to the form there are also the logos of the miner hosts. Beneath each of those logos, there are the status messages for the corresponding miner. If something goes wrong with the communication, the user can easily see which miner gave an error and report this.


Est | Eng | Sisukord | [redacted] x
Üldinfo
Liikmed
Uudised
Tegevus
IKT klaster
Koostöö
Kalender
Liikmetele
Eesti Infotehnoloogia ja Telekommunikatsiooni Liit

Ettevõtte esialgsete majandusandmete esitamine

Aasta:	2009
Käive (Eurodes):	
Töötajate arv (inimeste arv):	
Ekspordi osakaal (protsentides):	
Lisaväärtus* (Eurodes):	
Tööjõukulud (Eurodes):	
Koolituskulud (Eurodes):	
Puhaskasum (Eurodes):	

Koos kõigi maksudega, sh erisoodustus

Esita andmed

* lisaväärtus = tööjõukulud + amort + kasum + otsesed maksud.

CYBERNETICA
- Valmis

MicroLink
- Valmis

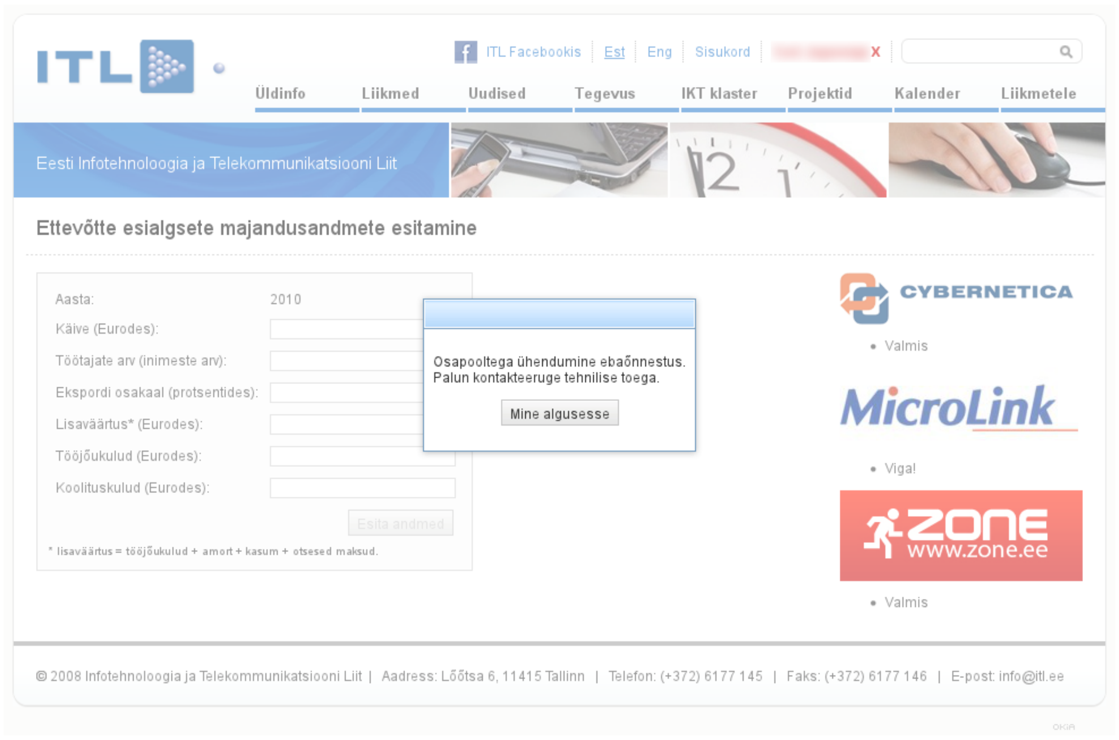
ZONE www.zone.ee
- Valmis

© 2008 Infotehnoloogia ja Telekommunikatsiooni Liit | Aadress: Lõõtsa 6, 11415 Tallinn | Telefon: (+372) 6177 145 | Faks: (+372) 6177 146 | E-post: info@itl.ee

Figure 8: ITL web page member area. The submission form.

Once the web page with the form is loaded, the JavaScript library in it starts to collect randomness from all of the miners as described in Section 5.3. When the randomness is successfully received from one of the miners, a status text “Ready” (“Valmis” in Estonian) appears beneath the corresponding miner host’s logo. After collecting randomness from all of the three miners and initializing AES with it, the “Send” button of the submission form that was initially disabled, is now enabled. However, if the JavaScript library fails to receive randomness from one of the miners, it adds a status text “Error” (“Viga” in Estonian) beneath the corresponding logo(s) and informs the user with a modal dialog that he or she cannot continue filling the form at the moment (see Figure 9).

If the randomness is successfully received from all three miners and the user has finished filling in the form, he or she can then press the (now enabled) “Submit” (“Esita andmed” in Estonian) button. Now all of the user-inserted values are parsed (from string to integer) and validated. For the economic indicators, all inserted values must represent non-negative integers, except for the field “Profit” (“Puhaskasum” in Estonian) which can also be a negative integer. The fields that did not pass validation, are colored pink so the user can identify them and correct the mistakes.



Ettevõtte esialgsete majandusandmete esitamine

Aasta:	2010
Käive (Eurodes):	
Töötajate arv (inimeste arv):	
Ekspordi osakaal (protsentides):	
Lisaväärtus* (Eurodes):	
Tööjõukulud (Eurodes):	
Koolituskulud (Eurodes):	

Esita andmed

* lisaväärtus = tööjõukulud + amort + kasum + otsesed maksud.

Osapooltega ühendumine ebaõnnestus.
Palun kontakteeruge tehnilise toega.

Mine algusesse

- Valmis
- Viga!
- Valmis

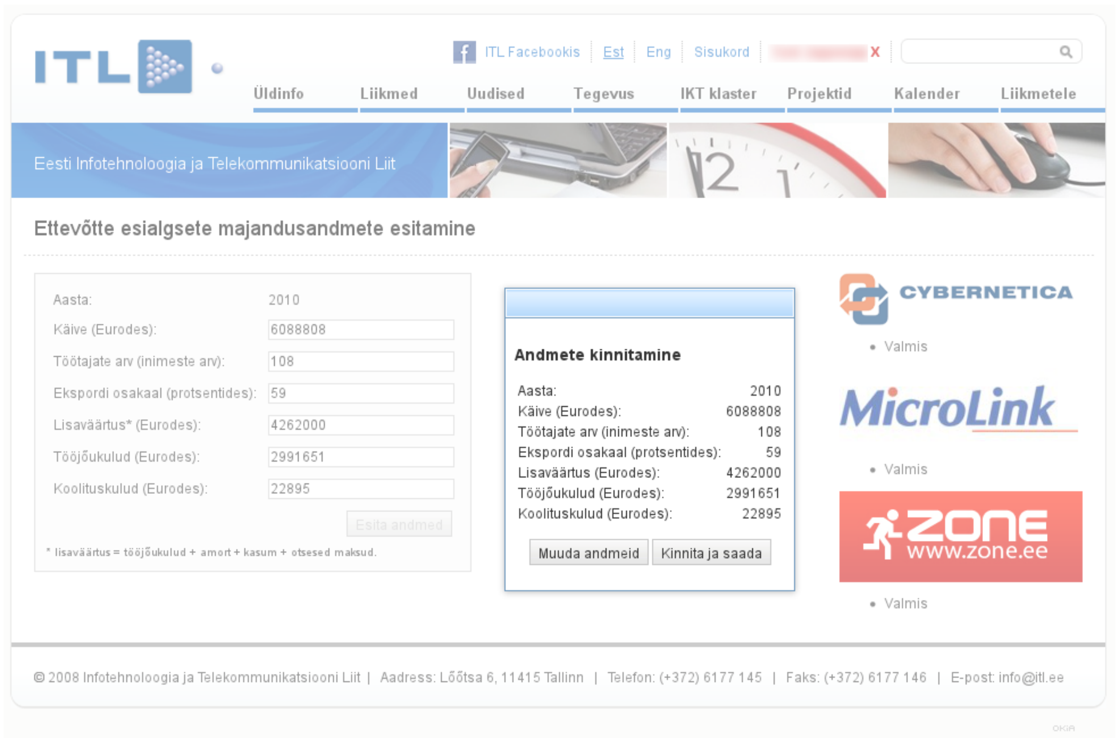
© 2008 Infotehnoloogia ja Telekommunikatsiooni Liit | Aadress: Lõõtsa 6, 11415 Tallinn | Telefon: (+372) 6177 145 | Faks: (+372) 6177 146 | E-post: info@itl.ee

Figure 9: ITL web page member area. One of the miners has encountered an error and the form cannot be submitted at this time.

If all fields pass validation, the user is presented with a modal dialog box (see Figure 10) where he or she can see all of the values inserted, however, the values cannot be changed here anymore. This confirmation is necessary for the user to make sure that all of the values were parsed correctly and there were no errors. The user can either click the “Confirm and send” (“Kinnita ja saada” in Estonian) button to secret share those values and distribute them among the miners or the “Change values” (“Muuda andmeid” in Estonian) button if he or she notices any mistakes and wants to go back and modify the inserted values.

6.3.2 Sending the data

Upon clicking the “Confirm and send” button in the confirmation dialog, all of the inserted values are secret shared and each share is sent to one of the miners along with some public data (randomly generated session identifier, company id). Also a text “Sending classified data” (“Saadan salastatud andmeid” in Estonian) is added to the status texts beneath each miner host’s logo. It is important to notice, that clicking the “Confirm and send” button does not actually submit the form to the origin server but rather executes a JavaScript function that performs


ITL Facebookis | Est | Eng | Sisukord | [redacted] X
Üldinfo
Liikmed
Uudised
Tegevus
IKT klaster
Projektid
Kalender
Liikmetele
Eesti Infotehnoloogia ja Telekommunikatsiooni Liit

Ettevõtte esialgsete majandusandmete esitamine

Aasta:	2010
Käive (Eurodes):	6088808
Töötajate arv (inimeste arv):	108
Ekspordi osakaal (protsentides):	59
Lisaväärtus* (Eurodes):	4262000
Tööjõukulud (Eurodes):	2991651
Koolituskulud (Eurodes):	22895

* lisaväärtus = tööjõukulud + amort + kasum + otsesed maksud.

Andmete kinnitamine

Aasta:	2010
Käive (Eurodes):	6088808
Töötajate arv (inimeste arv):	108
Ekspordi osakaal (protsentides):	59
Lisaväärtus (Eurodes):	4262000
Tööjõukulud (Eurodes):	2991651
Koolituskulud (Eurodes):	22895

- Valmis

- Valmis

- Valmis

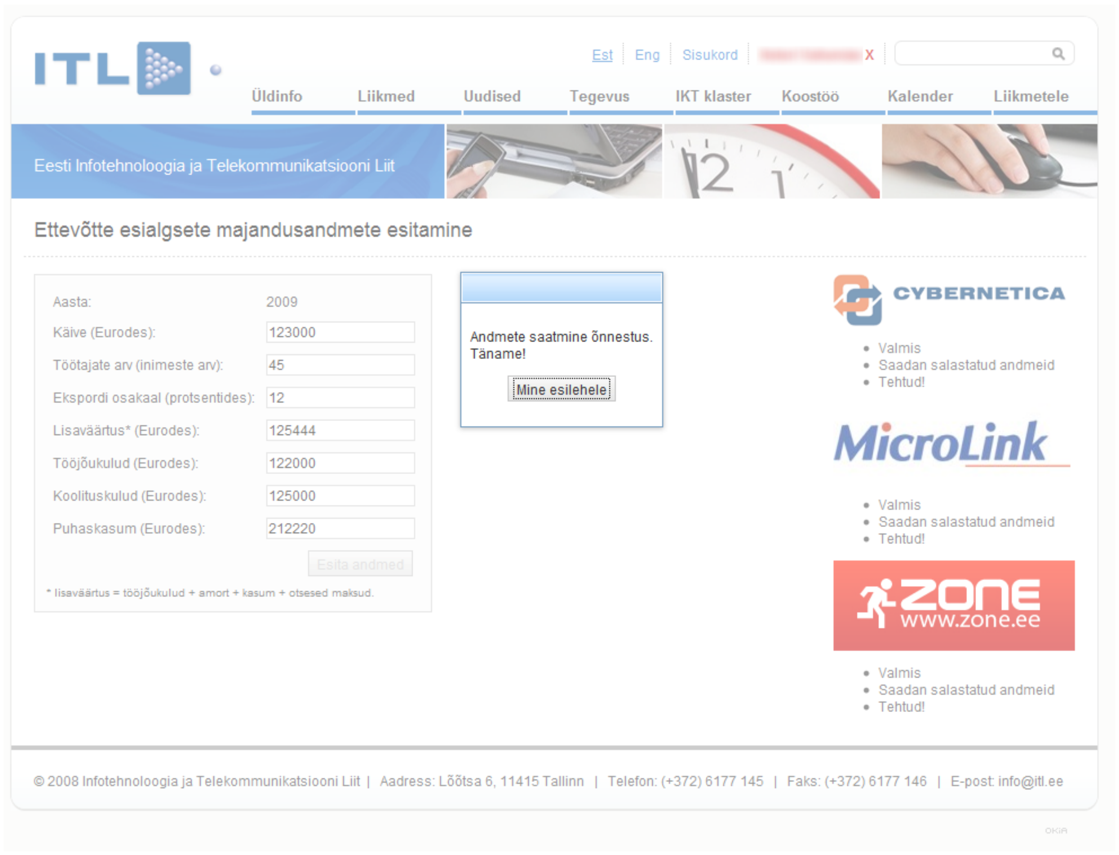
© 2008 Infotehnoloogia ja Telekommunikatsiooni Liit | Aadress: Lõõtsa 6, 11415 Tallinn | Telefon: (+372) 6177 145 | Faks: (+372) 6177 146 | E-post: info@itl.ee

Figure 10: ITL web page member area. Confirmation dialog showing all the inserted values.

secret sharing and distributes the shares among the three data miners as described in Section 5.4.

After that, when a success confirmation message is received from one of the miners, the text “Done” (“Tehtud” in Estonian) is added to the status texts beneath the corresponding logo. When the JavaScript library has received a positive acknowledgment from all of the miners, it also makes a query to the ITL web server notifying that this company has filled in the form. The user is then notified with a message that the submission has been successful (see Figure 11) and he or she is directed back to the page with the list of forms.

However, if sending the shares failed (e.g. a timeout was encountered) or one of the miners replied with an error message (e.g. the shares could not be saved in a database), a status text “Error” is added beneath the corresponding logo instead and the user is notified that the submission has failed. The user can then go back to the previous page.



Eesti Infotehnoloogia ja Telekommunikatsiooni Liit

Ettevõtte esialgsete majandusandmete esitamine

Aasta:	2009
Käive (Eurodes):	123000
Töötajate arv (inimeste arv):	45
Ekspordi osakaal (protsentides):	12
Lisaväärtus* (Eurodes):	125444
Tööjõukulud (Eurodes):	122000
Koolituskulud (Eurodes):	125000
Puhaskasum (Eurodes):	212220

* lisaväärtus = tööjõukulud + amort + kasum + otsesed maksud.

Andmete saatmine õnnestus. Täname!

- Valmis
- Saadan salastatud andmeid
- Tehtud!

- Valmis
- Saadan salastatud andmeid
- Tehtud!

- Valmis
- Saadan salastatud andmeid
- Tehtud!

© 2008 Infotehnoloogia ja Telekommunikatsiooni Liit | Aadress: Lõõtsa 6, 11415 Tallinn | Telefon: (+372) 6177 145 | Faks: (+372) 6177 146 | E-post: info@itl.ee

Figure 11: ITL web page member area. The data submission has been successful.

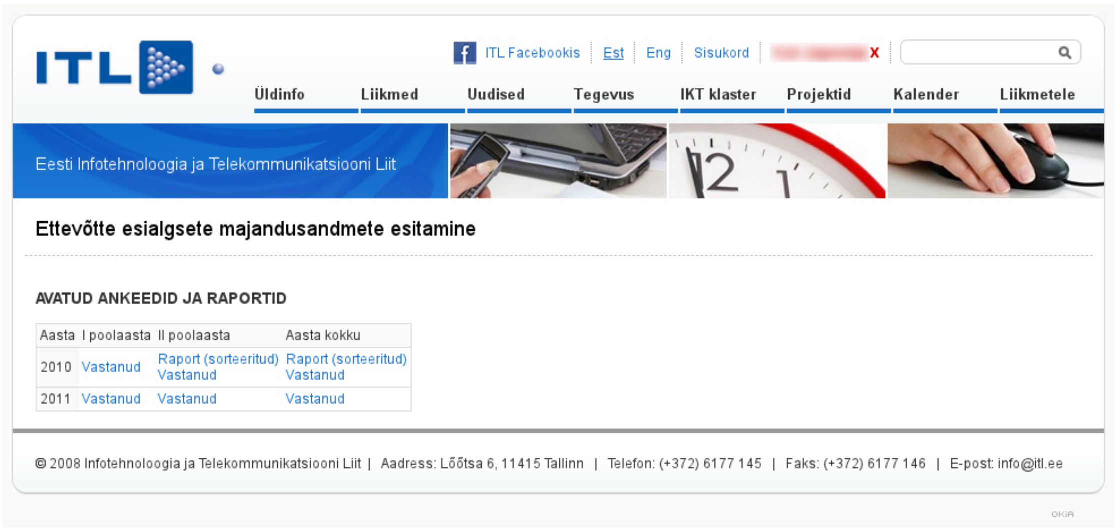
6.3.3 Submitting the same form twice

Since the freshness of the collected data is important, the data submission forms are opened only for a short period right after a year or half year has ended. Thus, the economic indicator values submitted by the companies may not yet have been audited and are subject to change. This means that the representative of a company may want to resubmit some economic indicators when he or she either receives some updated information or realizes that he or she has made a mistake while filling in the form.

However, since all of the inserted values are secret shared in the user's web browser and distributed among the three SHAREMIND miners, the representative of a company cannot see or edit the values he or she has previously submitted. So to correct some submitted economic indicators, the user has to refill and submit the corresponding form again. It is possible to refill and resubmit a given form any number of times while the form is opened. For each form and each company, only the latest submitted values are taken into account in the data analysis process.

6.4 Reporting

Some reporting capabilities were also integrated with the ITL web page member area. When an ITL board member logs in to the web page, he or she can see some additional information on the submission form listing page (see Figure 12). Board members can also see the list of people who have already submitted their data and those who have not by clicking the link “Answered people” (“Vastanud” in Estonian).



Ettevõtte esialgsete majandusandmete esitamine

AVATUD ANKEEDID JA RAPORTID

Aasta	I poolaasta	II poolaasta	Aasta kokku
2010	Vastanud	Raport (sorteeritud) Vastanud	Raport (sorteeritud) Vastanud
2011	Vastanud	Vastanud	Vastanud

© 2008 Infotehnoloogia ja Telekommunikatsiooni Liit | Aadress: Lõõtsa 6, 11415 Tallinn | Telefon: (+372) 6177 145 | Faks: (+372) 6177 146 | E-post: info@itl.ee

Figure 12: ITL web page member area. Board members can also see the list of companies who have submitted their data. This screenshot is taken after the data collection period has ended and the reports have been generated, so the links to submission forms are replaced with links to reports.

When the data collection period is over for some form, the analysis process for this form is executed as described in Section 4.4. For the first collection period it was done manually. Firstly, the proxy applications were synchronously executed on all three miners. After the shares had been copied to the miners’ internal databases, the analysis application was run, which started the secure multiparty computations between the SHAREMIND miners. However, in the future we would still like to automate the report generating process so that the proxy applications and the analysis application are automatically started right after the collection period has ended. As the data collection deadline was postponed several times for the first collection period (see Section 7.1 for details), we could not automate the process this time.

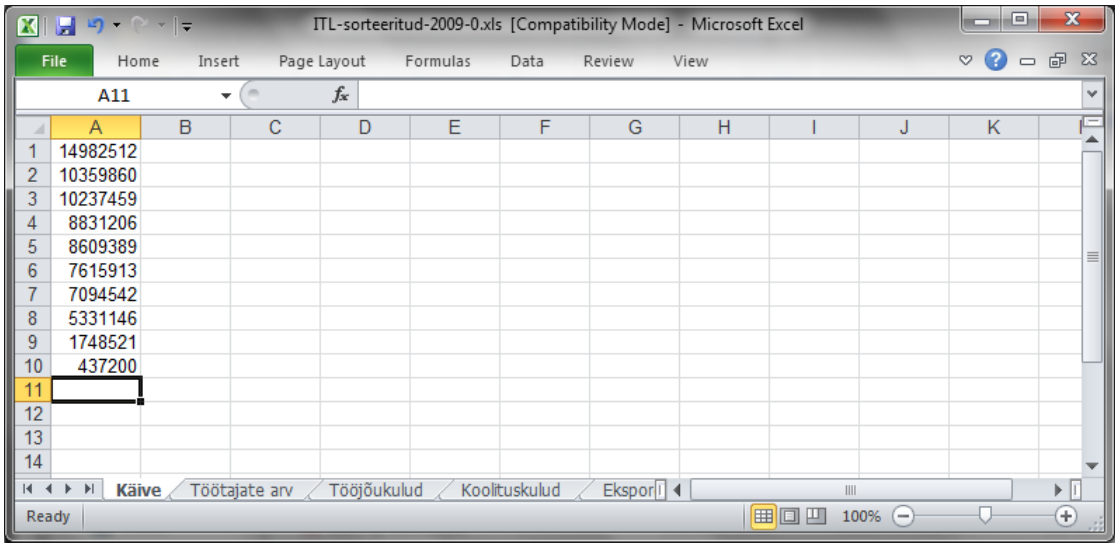
As stated in the initial requirements, the only analysis performed on the collected data is to sort each economical indicator independently. Each miner’s internal database has only one table that holds the shares from only one of the submission forms. This table has two columns for each indicator — one holding the share that represents the inserted value and another holding the share representing a boolean value of whether this indicator was actually inserted or not (i.e. should it be used in the analysis or not).

Each SHAREMIND miner has a copy of a SECREC script (see Appendix A) that takes the database table name and the two column names for a given indicator as parameters. Firstly, the script loads both vectors from the database and multiplies them value-by-value. This effectively replaces all of the values that were actually not inserted by zero. It is only a precaution as submitting all of the economic indicators is compulsory and thus all the values in the boolean vector should represent the value `true` (used as 1 in calculations). The multiplication result is a vector that is then passed to the secure bubble sorting algorithm implemented for the SHAREMIND framework by Uddin [25]. This sorting algorithm reorders the values in the data vector in decreasing order and works correctly even if the initial vector includes some negative values. Finally, the sorted vector is *published*, i.e. the original values are recombined from the shares and sent to the analysis application. The analysis application executes this SECREC program for each economic indicator and saves the sorted output vector in a database at the ITL web page.

According to the SHAREMIND deployment model (see Figure 4) the analysis application should be hosted by ITL, as this is the party controlling the data collection process. However, ITL uses virtual hosting service for its web page and it is impossible to install command line applications on that server. Thus, the analysis application was deployed in the SHAREMIND miner's host machine at Cybernetica.

Once the analysis application has saved the sorted indicators in the database, the board members see a new link "Report (sorted)" in the form listing page (see Figure 12). Clicking this link for any given form allows the board member to download a Microsoft Excel spreadsheet with all of the values entered by the members throughout the data collection period. This spreadsheet (see Figure 13) has all the inserted values, sorted independently by each economic indicator. The values for each indicator are shown on a separate worksheet to avoid confusion. If all of the indicators (columns) were on the same worksheet next to each other, it might give the impression that one row represents the data for a single company. However, since the indicators were sorted separately, this is not the case. Moreover, splitting the sorted vectors to separate worksheets also makes further data analysis (e.g. reordering) simpler.

Once the board members have gone over the spreadsheet and made the necessary enhancements (e.g. analyzed data and created charts) they can make the



	A
1	14982512
2	10359860
3	10237459
4	8831206
5	8609389
6	7615913
7	7094542
8	5331146
9	1748521
10	437200

Figure 13: Example report opened in Microsoft Excel. The economic indicator values shown here are randomly generated for illustrative purpose.

report available to other ITL members.

6.5 Training materials

To aid the users in filling in the submission form, we have created a user manual that describes all the necessary steps (see Appendix B, in Estonian). This manual includes an introduction where an ITL board member explains the background and motivation behind this sort of data collection. This is necessary to show other ITL members that the initiative comes within the ITL itself and collecting information about the economic indicators benefits all of the its members. Users are also notified about the privacy-preserving measures taken in this project, including the improvements achieved by using the SHAREMIND framework. This also allows the ITL members to further research the secure multiparty computation topic and learn about the privacy guarantees it gives.

There is also a guide for ITL board members on how to access the list of people who have submitted a given form and how to download the report (Microsoft Excel spreadsheet) (see Appendix C, in Estonian).

As this project is the first real life application of the SHAREMIND framework, we also recorded all the necessary steps how to install and configure a web server, SQL database engine and a SHAREMIND miner on a (virtual) server. This documentation (see Appendix D) will help system administrators to set up the necessary infrastructure if they want to host a miner in their domain.

7 Project lifecycle

7.1 Timeline

According to the initial schedule, the first data collection period should have been in the beginning of 2011, where there would be two open forms:

- the second half of 2010 (4 indicators), opened from Jan 10 to Jan 30;
- the whole year of 2010 (6 indicators[3]), open from Jan 10 to Feb 14.

However, ITL had also a new authentication method developed for their web site at that time and they preferred not to start the data collection period before this new feature had been fully deployed and tested. Finally the new authentication system was successfully deployed on Feb 4th. After that ITL decided to conduct the first data collection from Feb 9th to Feb 28th for both forms. They sent an invitation to a subset of 29 ICT companies from their members to submit their economic data on the ITL web page.

By the Feb 28th, only 7 companies had filled in the whole year form and 4 companies had filled in the form for the second half of 2010. Thus, ITL sent a new reminder to the companies and postponed the form submission deadline to Mar 4th. When the extended submission period ended on Mar 4th, there were 28 submissions in total by 17 different companies. This means that some of the companies had only submitted one of the two forms. After brief consideration, ITL board decided that the forms should be reopened for yet another week (Mar 10th - 18th).

Finally, the collection period was closed on March 18th. By this time, 17 companies had filled in the whole year form and 12 companies had filled in the form for the second half of 2010. On March 22nd we started the secure multiparty computation process and the resulting sorted economic indicators were published as a Microsoft Excel spreadsheet on the ITL web page for its board members.

However, ITL board did not forward the reports to ITL members right away. On April 13th a representative of the ITL board contacted us to ask if it would be possible to also calculate some ratios like added value divided by the number of employees, labor costs divided by the number of employees and training costs divided by labor costs. Since private division (i.e. division using secret shared data) is already implemented in the Sharemind framework, calculating the relative values can be easily accomplished by just dividing the two corresponding vectors.

[3]As 2010 was also the year of economic crisis, ITL decided not to ask companies to fill in the "profit" field for this period as it might be abnormally low for many companies. Thus, asking to fill this field could demotivate the companies from filling in the forms altogether.

Secondly, the representative of the ITL board was interested to see how the economic indicator values have changed between the periods of the submitted forms. Since we only have data from the second half of 2010 and from the full year of 2010, we can compare indicator values from these periods. However, it does not make sense to just compare the averages of those two periods as the number of companies who filled in the two forms are different. From the 17 companies who submitted the full year form and the 12 companies who submitted the form for the second half of the year 2010, only 11 companies filled in both of the forms. We can easily find those 11 companies as the company ID is present for each data row in the buffer databases. So for each economic indicator we have two vectors of 11 elements. To allow further analysis of the data but at the same time preserve its privacy, we just sorted the data vectors as in the initial report. We changed the `bubble_sort` SecreC function shown in Appendix A to preserve the data row associations between two vectors while sorting by the values of one vector. Of course, this kind of analysis will be more useful in the future when we have collected data for more than one year.

These two additional reports were sent to the ITL board on April 20th and on April 25th respectively. ITL board published them to all of the ITL members on April 28th.

7.2 User problems

After ITL board had published the final reports to its members, there was only one noticeable problem. A representative of one ITL member company could not find the value of labour costs he had entered from the report. However, he recognized all the other five values he had entered in that form and the ITL board verified that the company was also in the list on companies who had filled in the form for year 2010. We also made sure that the data row with the corresponding company ID was present in all three data miners' databases.

As this was the only reported case of missing data and the other five indicators entered by that company were present, we have no reason to believe that the secure multiparty calculations were faulty. Most probably the representative of the company had just forgotten the exact value he had entered in the data submission form.

To alleviate the risk of having this kind of problems in the subsequent data collecting periods, we plan to enable saving of the inserted values to the user's computer. After the user has successfully submitted the form, he or she is again presented with the inserted values so they could be saved to the disk.

7.3 User feedback

After publishing all the reports to its members, ITL board stated that it was pleased with the overall data collection and analysis process and would like to carry on collecting the economic indicators twice a year. We have already agreed on some new features (e.g. saving the inserted values to the user's computer) for the next data collection period. This time, we hope to get ITL members to fill in their data more quickly so we could stay in schedule and automate the data analysis process like it was initially planned. Based on the reports generated for the previous data collection period, ITL board will decide which kind of data analyses will be required for future reports.

ITL board sees this economic benchmarking data collection as a long term effort. When ITL has collected this data for a couple of years, we can generate extra reports including time series of different economic indicators. This data would be most valuable to the ITL members.

8 Conclusion

In this thesis we elaborate on a scenario where we have several companies working on the same field, who would like to collaborate to analyze the market in a whole. To run statistical analysis on their data, they would have to gather the data to one place. However, companies are reluctant to share their data as it provides them business value. Moreover, if their data includes some personal information of their customers, they are prohibited by the law to share it. In particular, we concentrate on the case of ITL — a consortium of IT companies in Estonia. ITL members are interested in the ICT sector analysis, but do not want to share their economic data.

We analyze the given problem and propose a solution using the SHAREMIND secure multiparty computation (MPC) framework. As the data collection is done using a web-based submission form and the SHAREMIND miners do not have a web interface, one of the main challenges is to perform secret sharing in the user's computer and distribute the shares securely among the data miners. For this reason we have developed a JavaScript library that takes care of collecting randomness and performing secret sharing in a web browser. Also we have built a proxy application that interfaces the SHAREMIND miner with a web server to receive shared data.

This implementation was successfully tested within ITL at the beginning of this year, when ITL collected the economic indicators for the year 2010 from its members. In this thesis, we have documented the timeline and all of the analysis done with the secret shared data. We have received positive feedback from the ITL board, stating that they would like to continue using the system we have deployed.

To the best of author's knowledge, the ITL application is the first real life application where secure multiparty computation techniques are used for joint data analysis. The success of this project has shown that secure MPC applications are indeed usable in solving real problems. With this kind of applications we are able to provide various organizations with data analysis results that are otherwise complicated or even impossible to acquire due to legal or organizational restrictions.

Juhtumianalüüs: turvalise ühisarvutuse rakendamine jagatud andmete analüüsimisel

Magistritöö (30 EAP)

Riivo Talviste

Resümee

Vaatleme olukorda, kus grupp samal alal tegutsevaid ettevõtteid soovib analüüsida oma tegevusvaldkonda. Igaühel neist on oma klientuur, kuid selle uurimisest üksi on vähe. Ettevõtete juhid on huvitatud terve valdkonna ja selle klientide analüüsist, et tulemuste põhjal ärilisi otsuseid langetada.

Kogu valdkonna analüüsiks tuleks vastavatelt ettevõtetelt nende tegevusnäitajad ühte kohta kokku koguda, et neid saaks korralikult analüüsida. Samas ei ole ettevõtted aga oma info jagamisest huvitatud, kuna see on üks osa nende ärisaladusest. Veelgi enam — kui see info peaks sisaldama ka näiteks ettevõtte klientide isiklikke andmeid, siis on sellise info jagamine teistele osapooltele seadusandlusega keelatud.

Sellise probleemi lahendamiseks on välja töötatud turvalised ühisarvutuse skeemid. Delikaatsete andmete privaatsuse säilitamiseks jagatakse kõik andmed esmalt osakuteks ning jagatakse mitme sõltumatu osapoole vahel laiali. Kuna iga osak näeb välja nagu juhuslik arv, ei saa üksi osapool oma valduses olevate andmete põhjal teha mingeid järeldusi algandmete kohta. Samas on aga mitme osapoole valduses olevaid osakuid kombineerides võimalik taastada algsed andmed. Ühissalastatud andmete peal on võimalik teha ka arvutusi, kasutades selleks turvalise ühisarvutuse protokolle. Nende protokollide abil on võimalik arvutada funktsiooni väärtusi ühissalastatud andmete peal nii, et kõik osapooled saavad teada funktsiooni lõpptulemuse, kuid mitte ühtegi vahepealse arvutuse tulemust.

Käesolev töö keskendub eelpool kirjeldatud probleemi ühele konkreetsele näitele. Infotehnoloogia ja Telekommunikatsiooni Liit (ITL) on mittetulundusühing, mis ühendab enam kui 60 info- ja kommunikatsioonitehnoloogia (IKT) alal tegutsevat ettevõtet Eestis. Kuna IKT on kiiresti arenev majandussektor, on need ettevõtted huvitatud ajakohaste majandusnäitajate võrdlemisest, et langetada paremaid strateegilisi otsuseid.

Et ITL saaks neid andmeid turvaliselt koguda ja analüüsida, pakkusime välja lahenduse, kus kasutame SHAREMINDi raamistikku. SHAREMIND on hajus andmebaasi- ja rakendusserver, mis hoiab andmeid ühissalastatud kujul ning kasutab turvalise ühisarvutuse protokolle tagamaks, et üksikud algväärtused andmeanalüüsi käigus ei lekiks. Ühissalastatud andmete hoidmiseks kasutab SHAREMIND kolme sõltumatut osapoolt ehk andmekaevurit.

ITL-i rakenduse puhul toimub majandusnäitajate sisestamine ITL-i veebilehel asuva andmesisestusvormi abil. Sel juhul peab sisestatud andmete ühissalastus toimuma kasutaja veebilehitsejas, kuna SHAREMINDi pakutava privaatsusgarantii toimimiseks ei tohi algkujul olevad andmed kasutaja arvutist lahkuda. Saadud osakud tuleb seejärel jagada kolme SHAREMINDi andmekaevuri vahel. Ka see osutub probleemiks, kuna praegusel andmekaevuri versioonil puudub veebiliides. Et tekkinud takistusi lahendada, kirjutasime JavaScripti teegi, mis tegeleb kasutaja veebilehitsejas andmete ühissalastusega ning tekkinud osakute saatmisega andmekaevuritele. Iga SHAREMINDi andmekaevuri ühendasime veebirakendusega, mis kasutajalt osakuid vastu võttis ning spetsiaalse vahendusprogrammi abil andmekaevurini toimetas.

Kirjeldatud lahendust kasutas ITL selle aasta alguses, et koguda oma liikmete majandusnäitajaid 2010. aasta kohta. Käesolevas töös on kirjeldatud nii kasutatud lahenduse tehnilised aspektid ja andmeanalüüs kui ka projekti ajaline käik. ITL-i juhatus on esimese andmekogumisperioodi tulemustega rahul ning soovib antud rakenduse kasutamist kindlasti jätkata.

Autori andmetel on väljatöötatud rakenduse näol tegemist esimese reaalsest elust pärit probleemi lahendusega, kus turvalist ühisarvutust kasutatakse jagatud andmete analüüsiks. ITL-i rakenduse edu näitab, et taoliste lahendustega saab ka tulevikus pakkuda organisatsioonidele selliseid andmeanalüüsi tulemusi, mille leidmine ilma turvalise ühisarvutuse tehnoloogiata oleks raskendatud või lausa võimatu.

References

[1] Personal data protection act. RTI, 16.03.2007, 24, 127; Published online at `http://www.legaltext.ee/text/en/XXXX041.htm`. Last visited on May 29, 2009.

[2] Isikuandmete kaitse seadus. 15.02.2007. - RT I 2007, 24, 127; RT I, 30.12.2010, 2, 2007.

[3] NIST SP 800-90, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, 2007.

[4] M. Barbaro and T. Zeller. A face is exposed for AOL searcher no. 4417749. *New York Times*, 9:2008, 2006.

[5] G.R. Blakley. Safeguarding cryptographic keys. In *Proceedings of the 1979 AFIPS National Computer Conference*, pages 313–317, Monval, NJ, USA, 1979. AFIPS Press.

[6] Dan Bogdanov, Sven Laur, and Jan Willemson. Sharemind: A framework for fast privacy-preserving computations. In *ESORICS*, pages 192–206, 2008.

[7] Peter Bogetoft, Dan Lund Christensen, Ivan Damgård, Martin Geisler, Thomas Jakobsen, Mikkel Krøigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael Schwartzbach, and Tomas Toft. Multiparty computation goes live. Cryptology ePrint Archive, Report 2008/068, 2008. `http://eprint.iacr.org/`.

[8] Peter Bogetoft, Ivan Damgård, Thomas Jakobsen, Kurt Nielsen, Jakob Pagter, and Tomas Toft. A practical implementation of secure auctions based on multiparty integer computation. In *Proc. of Financial Cryptography '06*, volume 4107 of *LNCS*, pages 142–147. Springer, 2006.

[9] J. Boyar. Inferring sequences produced by pseudo-random number generators. *Journal of the ACM (JACM)*, 36(1):129–141, 1989.

[10] Tønnes Brekne, André Årnes, and Arne Øslebø. Anonymization of ip traffic monitoring data: Attacks on two prefix-preserving anonymization schemes and some proposed remedies. In George Danezis and David Martin, editors, *Privacy Enhancing Technologies*, volume 3856 of *Lecture Notes in Computer Science*, pages 179–196. Springer Berlin / Heidelberg, 2006. 10.1007/11767831_12.

[11] Martin Burkhart, Mario Strasser, Dilip Many, and Xenofontas Dimitropoulos. SEPIA: privacy-preserving aggregation of multi-domain network events and statistics. In *Proc. of USENIX Security 2010*, USENIX Security'10, pages 15–15. USENIX Association, 2010.

[12] European Council. Directive 95/46 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. OJ L 281, 1995.

[13] Ronald Cramer and Ivan Damgård. Multiparty computation, an introduction. Course Notes, 2004.

[14] D. Crockford. RFC 4627: JavaScript Object Notation. 2006.

[15] The Dojo Foundation. The Dojo Toolkit. Published online at `http://www.dojotoolkit.org/`. Last visited on April 7, 2010.

[16] Roman Jagomägis. SecreC: a Privacy-Aware Programming Language with Applications in Data Mining. Master's thesis, Institute of Computer Science, University of Tartu, 2010.

[17] Hillol Kargupta, Souptik Datta, Qi Wang, and Krishnamoorthy Sivakumar. Random-data perturbation techniques and privacy-preserving data mining. *Knowledge and Information Systems*, 7(4):387–414, 2005.

[18] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *Proc. of IEEE S&P '08*, pages 111–125. IEEE Computer Society, 2008.

[19] B. Poettering. jsaes: AES in JavaScript. Published online at `http://point-at-infinity.org/jsaes/`. Last visited on April 7, 2010.

[20] Reimo Rebane. An integrated development environment for the SecreC programming language. Bachelor's thesis, Institute of Computer Science, University of Tartu, 2010.

[21] Adi Shamir. How to share a secret. *Commun. ACM*, 22:612–613, November 1979.

[22] Emily Stark, Mike Hamburg, and Dan Boneh. Symmetric Cryptography in Javascript. In *Annual Computer Security Applications Conference*, 2009.

[23] Riivo Talviste. Web-based data entry in privacy-preserving applications. Bachelor's thesis, Institute of Computer Science, University of Tartu, 2009.

[24] Riivo Talviste and Dan Bogdanov. An improved method for privacy-preserving web-based data collection. Cybernetica research report T-4-5, 2009.

[25] Abu Hamed Mohammad Misbah Uddin. Privacy Preserving Collaborative Anomaly Detection Using Secure Multi-party Computation. Master's thesis, Institute of Computer Science, University of Tartu, 2010.

[26] VIFF development team. The Virtual Ideal Functionality Framework. `http://viff.dk`, 2007-2011.

[27] W3C. HTML5 Web Messaging, W3C Working Draft 17 March 2011. Published online at `http://www.w3.org/TR/webmessaging/`. Last visited on May 21, 2011.

Appendix A SecreC source code for sorting data vectors

```
void main(public string table, public string value_column,
  public string submitted_column) {

  public int num_rows; num_rows = dbRowCount(table);

  stack_dbPushColumn(value_column, table);
  private int[num_rows] data_values;
  data_values = stack_popVector(num_rows);

  stack_dbPushColumn(submitted_column, table);
  private int[num_rows] data_submitted;
  data_submitted = stack_popVector(num_rows);

  private int[num_rows] data; data = data_values * data_submitted;

  private int[num_rows] sorted; sorted = bubble_sort(data);
  public int[num_rows] psorted; psorted = declassify(sorted);

  publish("sorted", psorted);
}

// Oblivious bubble sorting (in decreasing order)
// From Uddin's master thesis
private int[] bubble_sort(private int[] data) {
  public int n; n = vecLength(data);
  private int a; private int b;
  private bool comp;
  private int c; private int invc;
  public int i; public int j;

  for (i = n-1; i >= 1; i=i-1) {
    for (j = 0; j < i; j=j+1) {
      a = data[j];
      b = data[j+1];
      comp = (a > b);
      c = boolToInt(comp);
      invc = 1 - c;

      data[j] = a*c + b*invc;
      data[j+1] = a*invc + b*c;
    }
  }

  return data;
}
```

Appendix B User manual for ITL members

This document was sent to each ITL member company that was asked to submit its economic indicators. The user manual includes a short introduction with the motivation for collecting the economic indicators. The users are also informed about the Sharemind framework that is used for privacy-preserving data analysis. The main part of the manual is a user story that guides the user through the data submission process.

ITL majandusinfo kogumise süsteem

Andmekogumisvormi kasutusjuhend

Milleks andmeid kogume?

Eesmärk on anda juhtidele võimalus enda ettevõtte võrdlemiseks sektori parimatega. Mitte ainult sektori keskmisega, vaid ka parimatega!

Auditeeritud andmed on Äriregistris avalikud alles järgmise aasta teises pooles. Juhtimiseks, eriti dünaamilises IKT valdkonnas, on vaja infot aga kiiremini, isegi juhul, kui seetõttu info on ebatäielik või isegi natuke ebatäpne.

Kogume andmeid kaks korda aastas kiiresti peale poolaasta lõppu, küsime ainult 6 juhtimiseks kõige olulisemat numbrit ja töötleme andmeid väga turvaliselt. Loomulikult ei pruugi need numbrid olla auditeeritud, aga ülim täpsus ei olegi oluline.

Kogume andmeid ennenägematult turvaliselt

Mõistagi on enda majandustulemuste kiire avaldamine headele konkurentidele kõhedust tekitav. Appi tuli Cybernetica oma uudse konfidentsiaalsete andmete töötlemise tehnoloogiaga Sharemind (http://research.cyber.ee/sharemind/). Kogutavad andmed jagatakse sisestamise hetkel kolmeks osaks, mis eraldi vaadeldes näevad välja juhuslike väärtustena. Iga osa salvestatakse ühe andmekoguja juures (ITLi majandusinfo kogumisel on andmekogujateks Cybernetica, Microlink ja Zone Media). Andmeid töödeldakse turvalise ühisarvutuse protokollidega nii, et sisendväärtuseid ei näe ükski andmekogujatest.

Kui andmekogumine on lõppenud, sorteeritakse Sharemindi abil iga tunnus eraldi tabelisse ja anonümiseeritakse. Niimoodi valminud tabelid saadame kõigile ITL liikmetele.

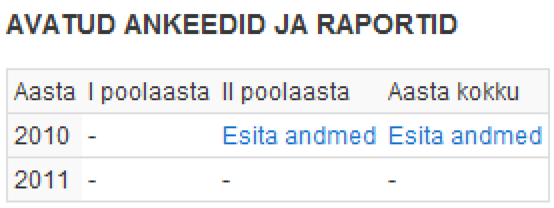
Vormi leidmine ITL-i veebilehel

1. Minge veebilehele www.itl.ee ning logige oma kasutajanime ja parooliga sisse.
2. Valige lehe ülemisest menüüst alajaotus *Liikmetele*.
3. Lehe vasakusse serva tekkinud menüüst valige *Majandusinfo kogumine*.
4. Valige *Andmete sisestamine.* Kui avaneb tühi leht, siis võib see olla põhjustatud sellest, et teil puuduvad lehele juurepääsuks vajalikud õigused. Palun võtke ühendust ITL-iga!

Andmete sisestamine

Andmeid ootame 45 päeva jooksul peale aasta ja poolaasta lõppu. Enne kui kõik on oma andmed sisestanud, kokkuvõtvaid tabeleid ei tehta. Seda selleks, et vältida anonümiseerimise *reverse engineeringut*.

Näide: juuresoleval pildil kujutatud olukorras on hetkel täitmiseks avatud 2010. aasta II poolaasta vorm ja kogu aasta vorm.



AVATUD ANKEEDID JA RAPORTID

Aasta	I poolaasta	II poolaasta	Aasta kokku
2010	-	Esita andmed	Esita andmed
2011	-	-	-

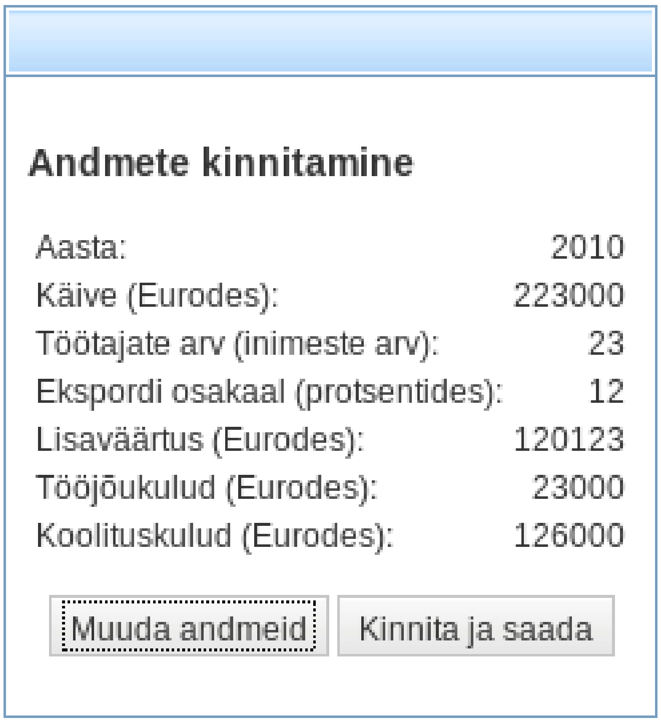
Valides mõne *Esita andmed* linkidest avaneb leht majandusandmete sisestamiseks vajaliku vormiga. Poolaasta vormil on vaja täita neli välja ja aasta vormil seitse. Tasub tähele panna, et sisestada saab ainult täisarve **eurodes** ning protsentuaalsete andmete puhul (näiteks *ekspordi osakaal*) peab sisestatud arv jääma vahemikku 0 – 100.

Lehe parempoolses servas on kolme konfidentsiaalse andmetöötluse eest vastutava koostööpartneri (Cybernetica, Microlink ja Zone Media) logod ning serverite olekuteated. Mõned sekundid peale lehe laadimist peaks iga logo alla tekkima kiri „Valmis“, mis tähendab, et antud serveriga on ühendus loodud. Kui ühendus on loodud kõigi kolme serveriga, siis aktiveeritakse ka vormi all olev nupp *Esita andmed*, mis alguses oli mitteaktiivne.

Juhul, kui mõne logo alla tekib hoopis kiri „Viga“, siis ei saadud vastava serveriga ühendust ning majandusandmeid ei ole hetkel võimalik esitada. Sel juhul palun saatke antud lehest tehtud ekraanitõmmis süsteemi tehnilise toe aadressile. Aadressi leiate juhendi lõpust.

Kui viga ei tekkinud ning kõik vormi väljad on täidetud, võib vajutada nupule *Esita andmed*. Kui mõnele väljale sisestatud arvud on vigased, siis läheb vastav väli roosaks. Sel juhul tuleb tuleb vastav väärtus korrigeerida ning uuesti vajutada nupule *Esita andmed*.

Kui kõik väljad on korrektselt täidetud, ilmub ekraani keskele andmete kinnitamise aken (vt. kõrvalolevat joonist), kus on näha kõik sisestatud väärtused. Vajadusel võib teha vormis korrektuure, vajutades nupule *Muuda andmeid*.


Andmete kinnitamine

Aasta:	2010
Käive (Eurodes):	223000
Töötajate arv (inimeste arv):	23
Ekspordi osakaal (protsentides):	12
Lisaväärtus (Eurodes):	120123
Tööjõukulud (Eurodes):	23000
Koolituskulud (Eurodes):	126000

Andmete esitamiseks vajutage nupule *Kinnita ja saada*. Selle peale peaks kõigi kolme osapoole logo alla lisanduma teade „Saadan salastatud andmeid“ ning seejärel „Tehtud!“ Peale seda ilmub ekraani keskele aken teatega „Andmete saatmine õnnestus.“ ning te võite tagasi pöörduda majandusinfo kogumise esilehele, vajutades nuppu *Mine esilehele*.

Juhul kui teadete „Saadan salastatud andmeid“ või „Tehtud!“ asemel ilmub mõne logo alla kiri „Viga“, katkestatakse andmete saatmine ning ilmub teade „Andmete saatmine ebaõnnestus.“ Sel juhul saatke jällegi antud lehest tehtud ekraanitõmmis tehnilisele toele.

Kui teil on küsimusi

Kui teil on küsimusi andmekogumise eesmärkide kohta, siis neid aitavad selgitada ITL projektijuht */nimi ja kontaktandmed eemaldatud/* ja ITL juhatuse liige */nimi ja kontaktandmed eemaldatud/*. Tehniliste küsimuste või vigade tekkimisel aitavad teid */nimi ja kontaktandmed eemaldatud/* ja */nimi ja kontaktandmed eemaldatud/*.

Appendix C User manual for ITL board members

This document was created for the ITL board members. It describes the reporting capabilities built into the ITL web page member area.

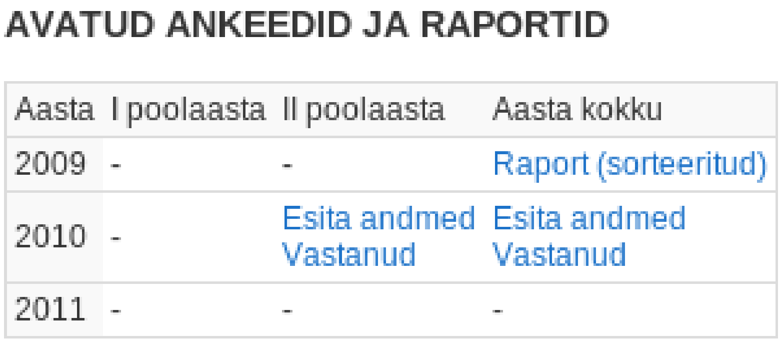
ITL majandusinfo kogumise süsteem

Andmekogumisvormi kasutusjuhend juhatusele

Vormi leidmine ITL-i veebilehel

1. Minge veebilehele www.itl.ee ning logige oma kasutajanime ja parooliga sisse.
2. Valige lehe ülemisest menüüst alajaotus *Liikmetele*.
3. Lehe vasakusse serva tekkinud menüüst valige *Majandusinfo kogumine*.
4. Valige *Andmete sisestamine*. Ekraanile peaks ilmuma sarnane tabel, nagu kujutatud allpool oleval joonisel. Kui avaneb tühi leht, siis võib see olla põhjustatud sellest, et teil puuduvad lehele juurepääsuks vajalikud õigused. Palun võtke ühendust ITL-iga!

   Antud tabelis kujutatud linke *Vastanud* ja *Raport (sorteeritud)* näevad ainult ITL liikmed, kes kuuluvad gruppi *Juhatus*.


AVATUD ANKEEDID JA RAPORTID

Aasta	I poolaasta	II poolaasta	Aasta kokku
2009	-	-	Raport (sorteeritud)
2010	-	Esita andmed Vastanud	Esita andmed Vastanud
2011	-	-	-

Vastanute nimekiri

Valides lingi *Vastanud*, avaneb leht, kus on kaks nimekirja – need on on vastava aasta/poolaasta andmed juba esitanud ja need, kes seda veel teinud ei ole. Iga inimese nime järel on välja toodud ka tema e-posti aadress, et oleks mugav meeldetuletusi saata.

Nendes nimekirjades ei ole mitte kõik ITL liikmed, vaid ainult need, kes peaks oma majandustulemusi selle süsteemi kaudu teatama – s.t. inimesed, kes kuuluvad gruppi *Majandusinfo*.

Raportite allalaadimine

Lõppenud andmekogumise perioodide kohta, mille vastav vorm on suletud ja tulemused arvutatud, on võimalik alla laadida raportit. Klikkides lingile *Raport (sorteeritud)*, pakutakse allalaadimiseks (või laetakse kohe alla – olenevalt teie veebilehitseja seadistusest) vastava perioodi raportit.

Raport ise on Exceli fail (.xls formaadis), kus iga kogutava tunnuse kohta on eraldi tööleht. Igal töölehel on ainult üks tulp – kõik sellel andmekogumise perioodil sisestatud antud tunnuse väärtused sorteeritud kahanevas järjekorras.

Kui mõnel perioodil on andmeid sisestatud ja andmekogumise periood on lõppenud, aga vastava raporti allalaadimise linki pole, siis tähendab see, et kogutud andmeid pole veel jõutud töödelda (sorteerida). Kui see saab tehtud, tekib ka vastav raporti link tabelisse.

Appendix D Sharemind miner host setup manual

Introduction

This document describes how to set up a working Sharemind miner host on your server. This process includes installing a Sharemind miner, a proxy application and web-based front-end for the miner. This document also briefly describes how to configure the web server and its SSL support.

As all Sharemind applications are unique and have a different data model, we use the private survey application as an example throughout this document. A live version of this survey is accessible at `http://sharemind.cyber.ee/survey/promo/index.html`.

Setting up Sharemind miner host

Installing and configuring a Sharemind miner

Prerequisites

- Sharemind source code or access to its SVN repository
- development tools (g++, make)
- libraries (refer to `docs/src/BUILD-General.dox` in Sharemind source code root folder for a list)
- information about other two miner hosts – their domain names, IP addresses, administrative and/or technical contact

Install Sharemind miner

1. extract given source code archive:
   ```
   tar xzf sharemind-release.tar.gz
   ```
2. `./configure --prefix=/path/to/install`
3. `make install`
4. if you use firewall, open UDP port for miner:
   ```
   iptables -A INPUT -p udp -m udp --dport 30003 -j ACCEPT
   ```

5. generate a key pair for the miner using:
   `./KeyPairGenerator --private-key minerX-private-key --public-key \`
   `minerX-public-key`

6. send your `minerX-public-key` to two other miner hosts

7. edit your miner configuration file and its whitelist

Set up proxy application

1. edit the database and table names in
   `contrib/webcontroller/src/WebControllerProxy.cpp`

2. run —make— in `contrib/webcontroller` copy `WebControllerProxy` and `proxy1.cfg` from `bin` folder to your installation `bin` folder

3. generate keys for your proxy application
   `./KeyPairGenerator --private-key proxyX-private-key --public-key \`
   `proxyX-public-key`

4. send the `proxyX-public-key` to other miner hosts

5. if you use firewall, open UDP port for proxy:
   `iptables -A INPUT -p udp -m udp --dport 30004 -j ACCEPT`

6. edit your proxy configuration file

Setting up a web server

Prerequisites

- Web server with PHP module (running as module or CGI)
- PHP >= 5.2 with mysqli module enabled
- MySQL server and client
- Access to a MySQL database
- URL of the final web-based survey application
- Access to the SVN repository of web front-end scripts or an archive containing those files

Configure SSL support

The web server should support HTTPS (HTTP over SSL) protocol. If you do not have a CA of your own, then one of the simplest ways to get widely accepted free SSL certificate is to acquire one from StartSSL [`http://www.startssl.com`]. StartSSL free ertificates are by default accepted by most modern web browsers.

To be able to authenticate yourself to StartSSL, you should be able to access either postmaster, hostmaster or webmaster e-mail accounts for the domain you are trying to get the certificate.

Once you get the certificate, you should install it in your web browser. Foe example, there is a tutorial on how to do it for Apache: `https://www.startssl.com/?app=21`

Miner web front-end

1. extract the `miner` folder from the provided private-survey archive and go to that folder:
   ```
   cd miner
   ```

2.
   ```
   ./create-miner-db-script.sh TABLE_NAME > table.sql
   ```

3.
   ```
   mysql -u USERNAME -p DATABASE < table.sql
   ```

4.
   ```
   cp -R export www /path/to/your/web/folder
   ```
5. edit `/path/to/your/web/folder/config.php`
   Change the miner ID 1,2,3 and MySQL connection information. Table name is the table you just created.
6. edit `/path/to/your/web/folder/message_passing.html`
   Change the row
   ```
   var parentHost = 'http://localhost';
   ```
   to use the real hostname and protocol of the final web-based application.
7. test if `random.php` in the `/path/to/your/web/folder/` is accessible from the web. It should show something like:
   ```
   {"miner":1,"random":"4c c4 02 90 99 6e 70 a1 00 25 55 bc 47 74 14
   50 5f 46 69 30 53 df 2f a4 5e 27 e9 3c fc 2e c3
   f5","success":true,"errno":0,"error":""}
   ```
8. if you use suhosin with PHP, then please edit your `suhosin.conf` (usually in `/etc/php5/conf.d`) to allow longer GET parameter values in requests:
   ```
   suhosin.get.max_value_length = 2048
   ```

Appendix E Source code of the JavaScript controller library