Mediating updates using a web based service

ZHENGJIAN ZHAO

KTH Information and
Communication Technology

Master of Science Thesis
Stockholm, Sweden 2007

COS/CCS 2007-18

Mediating updates using a web based service

Master Thesis Report

Zhengjian Zhao

zzha@kth.se

Academic Examiner & Supervisor:
Prof. Gerald Q. Maguire Jr.
School of Information and Communication Technology

Business Supervisor:
Doctor Sten Öhman
Antivenena AB

Abstract

The European Commission had approved the Registration, Evaluation and Authorization of Chemicals (REACH) regulations in the year 2006, and will begin to implement it form 1 June 2007. This regulation greatly increases safety of protection for the users of chemicals, but it also brought a problem for the manufactures as well as downstream users. That is they have to generate and distribute much more Material Safety Data Sheets (MSDS) then before. This work is difficult for most companies since this job usually was done manually. The new regulation requires a much more efficient method to generate and distribute them.

This thesis project addresses the problem of automatically distributing MSDS and the corresponding metadata. It presents the underlying technology selected for this project. It gives a brief introduction of this project, the underlying technologies used, along with the methods used to deliver relevant and up to date materials safety information.

At last an HTTP web application approach was selected to solve the problem, the resulting application can efficiently notify a downstream user of the newly updated MSDS and transport the corresponding file and metadata. It is truly data driven, therefore the downstream user does not need to check for updates everyday, instead will be notified when there is an available update.

Sammanfattning

Den Europeiska kommissionen antog 2006 en ny bestämmelse för informationsspridning av kemikalier, Registration, Evaluation, and Authorization of Chemicals (REACH), och den kommer att tas i bruk från första juni 2007. Bestämmelsen innebär ett starkt ökat skydd för användare av kemikalier, men den ställer även till problem för både tillverkare och användare. Exempelvis, de måste generera och distribuera betydligt fler materialsäkerhetsföreskrifter, Material Safety Data Sheets (MSDS), än tidigare. Detta arbete är krävande för de flesta företag eftersom arbetet ofta sker manuellt. Den nya bestämmelsen kräver mycket effektivare metoder att generera och distribuera MSDS.

Det här examensarbetet fokuserar på problemet att automatiskt distribuera MSDS och den korresponderande metadatat. Rapporten presenterar den underliggande teknologin för examensarbetet. Därutöver ges en kort introduktion till examensarbetet, den underliggande teknologin, tillsammans med de metoderna som används för att skicka relevanta och aktuella materialsäkerhetsföreskrifter.

Som avslutning väljs en HTTP Web applikationslösning för att lösa problemet. Den lösningen kan effektivt underrätta en användare att det finns en nyare MSDS och sedan skicka den korresponderande filen och dess metadata. Den är data driven, vilket gör att en användare inte måste kontrollera för uppdateringar varje dag, utan kommer att bli informerad när det finns en ny tillgänglig.

Acknowledgements

I would like to express my special thanks and most sincere gratitude to my academic supervisor and examiner Professor Gerald Q. Maguire Jr., for his time, his strong support, his comments on all my thesis report drafts and his thorough correction of my grammar.

I would also like to express my special thanks to my industrial supervisor Doctor Sten Öhman, for his encouragement throughout this thesis project, his kind advice and his time. I would also like to thank him for giving me the opportunity to do my thesis project at Antivenena.

Table of Contents

Chapter 1: Introduction

The project "REACH Automatic Safety Information" is a thesis project being conducted at Antivenena AB. The main aim of this project is to provide a safe and efficient Material Safety Data Sheet (MSDS) updating service to downstream users. Details of MSDS will be given in section 1.1.2.

1.1 Background

The European Commission approved the Registration, Evaluation and Authorization of Chemicals (REACH) regulations in the year 2006, and will begin to implement it from 1 June 2007. This regulation greatly increases the safety of the users of chemicals. It requires that all the manufactures and importers of chemicals provide adequate safety information for downstream users, so that these users can work properly with this chemical, so they could also take timely and appropriate actions when a hazard exists. The regulation also establishes a Chemicals Agency that will act as the central point of the REACH system. This agency will run a database in which consumers and professionals can find hazard information efficiently. [1]

1.1.1 REACH

REACH is aimed to improve the protection of human health and the environment through the better and earlier identification of the properties of chemical substances. The European Commission proposed the latest version on June 27 2006 [12]. It gives greater responsibility to industry to manage the risks from chemicals and to provide safety information on the substances, which they produce of distribute. Manufacturers and importers will be required to gather information on the properties of their substances, which will help the users manage them safely, and to register the information in a central database. A Chemicals Agency will act as the central point in the REACH system: it will run the databases necessary to operate the system, co-ordinate the in-depth evaluation of suspicious chemicals, and run a public database in which consumers and professionals can find hazard information. Further you can also get more information about the EU REACH legislation is available from reference [13].

1.1.2 MSDS

MSDS is designed to provide proper procedures for handling or working with a particular material for workers and emergency personnel. It contains information

about this material, such as physical data (melting point, boiling point, flash point, etc.), toxicity, health effects, first aid, reactivity, storage, disposal, protective equipment, and spill/leak procedures. These are very useful if a spill or other harmful accident occurs.

MSDSs are useful for employees who may be exposed to a hazard at work or employers who need to know the proper methods to store, dispose of material, etc; as well as emergency responders such as fire fighters, hazardous material crews, emergency medical technicians, and emergency room personnel.

Further information about MSDSs, see references [14], [15], and [11] (US style) and the second part of reference [12] (EU style).

Today most MSDSs are in Adobe's Portable Document Format (PDF); however other file formats are also used. One of these, XML, is strongly encouraged and under development [16] since it is quite easy to operate on the data and it is system independent, thus it is very possible that some web based distribution systems [17] will use this format in the future. Reference [18] is an example MSDS in HTML and PDF format.

1.1.3 What Others Have Done

Lots of companies focus on the creation, management, and distribution of MSDSs. Conseq Limited, a UK company [19] developed a system for Albemarle Corporation to manage the creation of master MSDSs and manage the workflow for MSDS control. The most interesting feature of the system which they developed is their support for automatic translation, which can generate MSDSs in many other languages.

Another company, The Wercs Ltd, declares that they understand the creation, management, and distribution of MSDS better than anyone in the industry [20], based the claim of 20 years of experience working with MSDSs.

In addition to these firms which are competing to create, manage, and distribute MSDSs, there are lots of manufactures and others for whom it is important to make sure that their latest MSDSs reach the relevant workers in order to migrate hazards. It is for this reason that we are designing and developing a system, to provide efficient high quality MSDS distribution. This MSDS distribution system should enable Antivenena's customers to update their MSDS database as soon as a newer version of any relevant MSDS is available.

1.2 Overview of master thesis

1.2.1 Problem Description

The problem solved in this project is to introduce an automatic distribution system for MSDSs between the manufacture and their downstream users. These downstream users need to receive the relevant MSDSs and store them in a local database (which acts as their chemical registry). They need these documents so that they can print or view them when necessary. However, today, most of this updating is done manually. This is a time consuming job since most MSDSs will be updated every other year and a given company may be working with many chemicals, hence they have many MSDSs to manage.

Antivenena AB collects information about risks and other properties of chemicals from open information sources and, after careful evaluation of its quality, then include this information into a database. This database is available for their customers using a LAMP (see section 3.2.1) web service (Trade mark: KemRisk).

In addition to information collected by Antivenena, KemRisk can also generate new MSDSs and store MSDSs received from manufactures. In the latter case important metadata about each MSDS will be included into the database to manage them easily, sine lots of MSDSs will be received, and the MSDS information should be stored in the database. The following items will be included in the metadata:

1. The name of the product.
2. The manufacturer's article number.
3. The identity of the manufacturer.
4. The edition date of the document.
5. Earlier edition dates that shall be replaced.
6. The R-Phrases of the product.
7. The S-Phrases of the product.
8. The flashpoint of the product.
9. Other important risk information.
10. Other important safety information.
11. The intended use(s) of the product.

These metadata cannot automatically be extracted from the MSDS and consequently this job is time-consuming and tedious. The latter is not specific to Antivenena's chemical registry system, but it is a common problem for all downstream users.

The goal of this project is to develop an automatic MSDS service by which the manufacturer can send a new or updated MSDS as well as the metadata related to this MSDS. Each MSDS and its metadata will be stored by the MSDS service. Subsequently this information is automatically transmitted to all registered downstream users of the product. The end user's chemical registry system (e.g. KemRisk) will update the local copy of the information (this local information

includes both the file and its metadata). Note that these users must subscribe to this distribution system in order to automatically receive the relevant updates. The whole process is shown in figure 1.

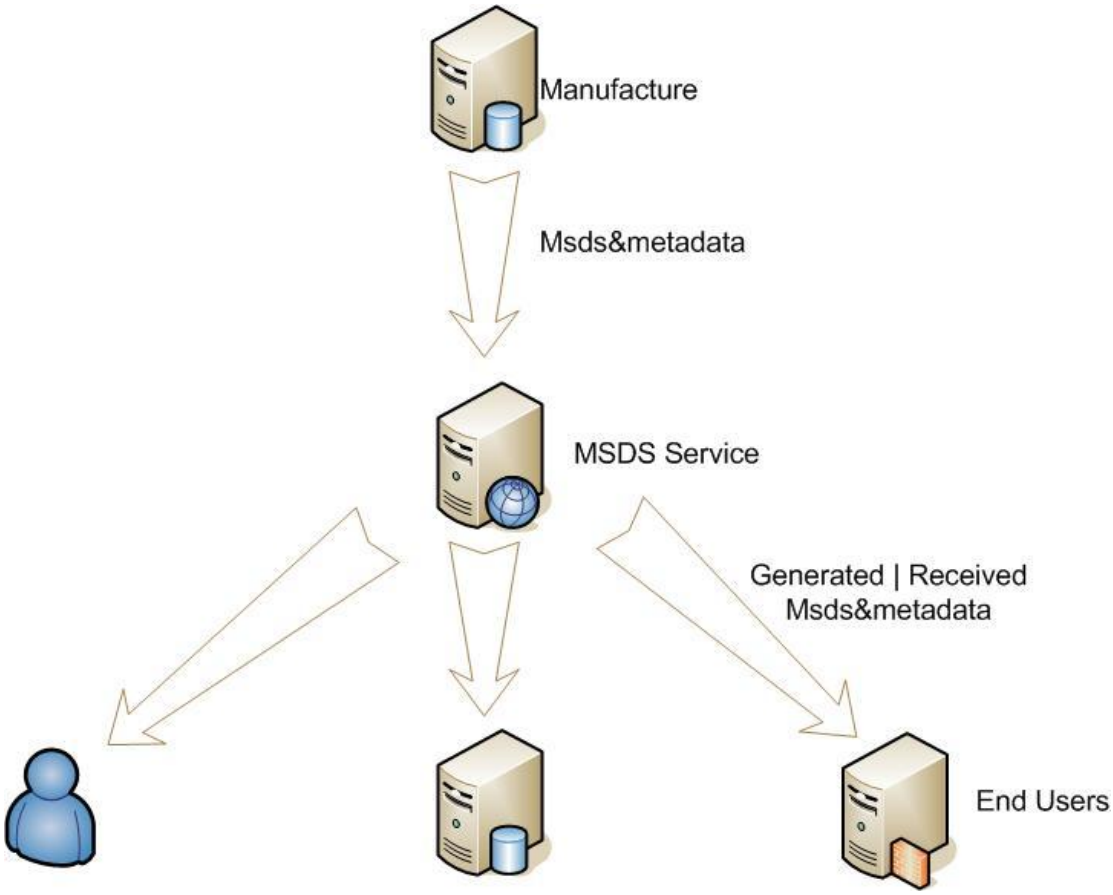

Figure 1 Problem Overview

1.2.2 Technical Goals

Basically, the project should achieve the following (measurable) technical goals:

1. Define a database structure for the following information:
   a) Information about each manufacture, including whether they have paid for this service
   b) Information about each downstream user of the chemical products, including whether they have paid for this service
   c) Information about each MSDS including the key data and where to get this MSDS.
2. A system between the manufacture and the MSDS service, so that the manufacture can inform the MSDS service of new versions MSDSs and send the key information and the updated MSDS to the MSDS service automatically and efficiently. This system should synchronize the data and the delay should not be very long for each MSDS. This communication system

will be referred as SysM in the following text.

3. A system between the MSDS service and the downstream user, so that the MSDS service can inform the downstream user that there is a newer version of a certain MSDS, and update it automatically. It should synchronize and if it a user does not update a given MSDS within a configured period, both the administrator of MSDS service and downstream user should be informed. This communication system will be referred as SysD in the following text.
4. Both the systems (SysM and SysD) should provide secure communication, both authenticating the parties involved and encrypting the actual communication session.
5. SysD should be fast enough to ensure all the downstream user can receive the update within one hour, averagely, it means the system should be able to transfer a 2M MSDS file to 600 customers within one hour
6. SysD should be independent of operating system of the downstream user.
7. All important events should be logged in an easy to audit and check format.

Chapter 2: Potential Technical Solutions

2.1. Database

The data structure of the database is very important, since it is very difficult to modify after a system is developed and large quantities of data are stored in the tables, so it is necessary to consider this structure carefully. Antivenena's existing database already has most of the tables which are needed to fulfill our requirements. However, we need to make some slight changes to them.

The existing tables that can be used in the project are described in table 1:

Table Name	Description
ActiveArt	Active product with packets size information
ActiveProd	Actives products
Companies	Companies
CountryTexts	Country text
LangsInCountries	Languages used in the countries
LanguageTexts	Language text
Msds	Msds information
Organizations	Organizations
SniTexts	List of business
TradeMarks	Trademarks
ValidCountries	Valid countries
ValidLangs	Valid languages

Table 1 Description of Existing Tables

The only function we lack in the tables are the indication of the company is a distributor/manufacture or a downstream user, thus we add two fields UpService_B and DownService_B to indicate their status, along with a date type field --ValidUntil, indicating this subscriber's payment status, if their subscription has expired no service should be provided. As some companies can both be a manufacture and downstream user, we need to add another ValidUntil field to indicate its payment status for its second role.

For some of the solutions described below, we may need to add additional information to the table "Organizations".

2.2. SysM & SysD

2.2.1 Pull method and push method

The SysM and SysD can be regarded as similar, i.e. they are both systems used to transfer metadata and files between servers. Therefore we discuss them together here.

In the implementation of data transfer between servers, two types of methods are possible to use. They are *pull* and *push* methods. Pull methods means the downstream server takes the initiative to "pull" data from the upstream server and push method means the upstream server takes the initiative to request the downstream server to accept information from it. As shown in figure 2. Note here I used "downstream server" and "upstream server" to indicate the server receiving data and the server providing data in both SysM and SysD, we will use the same notation when we discuss SysM and SysD in the following sections.

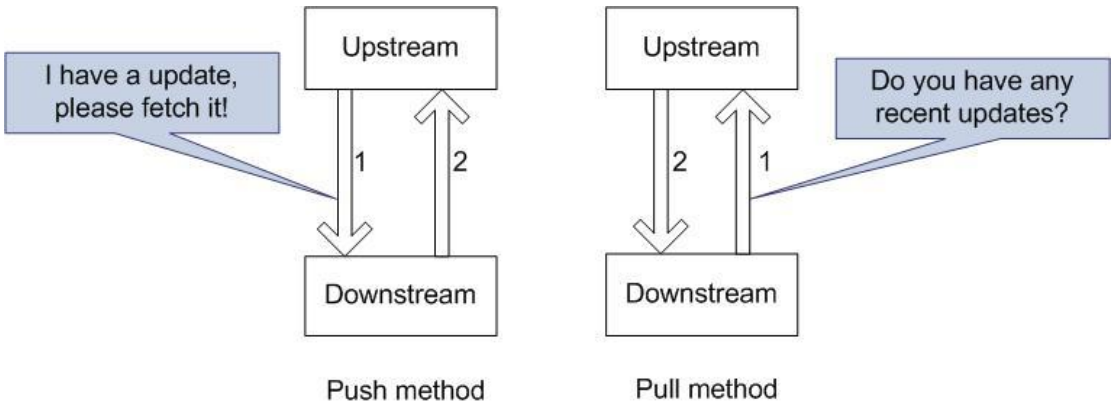

Figure 2 Push method and Pull method

In our case the push method is much preferred, since we can best ensure that the downstream server receives the transferred data as soon as there is an available update. Note that means the push method is in essence a data driven.

The push solutions can be divided into three main categories: HTTP web applications, mail solutions, and special purpose socket based server-client application. We will discuss each of them in the following sections. Note that we don't consider a distributed database server approach as this would assume that the various entities operate the same type of database, organize it in the same way and are willing to give others permission to make updates to it—since all these seem unlikely to be true, we have explicitly rejects such solutions and will not consider them farther.

2.2.2 HTTP web application solution

Assumption: assume that the upstream and downstream entities both have an

Apache server (see section 4.2.1) and an event logging system and the upstream server has either a database into which a trigger can be inserted or a daemon which they run to perform updates, a local email system, and an event logging system. Additionally the manufacturer is assumed to be willing to run a program which can acts as a web client to transfer the metadata and MSDS data to the MSDS service. This problem only exists when the upstream server is the manufacturer, since we assume that Antivenena is operating the MSDS service (thus they are assumed to be willing to run the proposed software).

Aim: to update the downstream server's database when there is an update to the upstream server's database. When an update occurs the manufacturer must send the MSDS and metadata together while updating the MSDS service database. Note this is push method, since it is driven by the update of the MSDS.

Solution: given an Apache HTTP server, we can use a server side script to finish the updating task. We use a daemon or a (mysql) database trigger on the upstream server's database, when an available update is detected, the trigger/daemon will send a request to the downstream user's Apache server, to trigger the downstream server to start its server side script, and the script on both downstream/upstream server should do the following tasks: 1. Authenticate the validity of the requested server. 2. Ensure the security of data transferring process. 3. Transfer the corresponding file and metadata. 4. Update the database. 5. Log this event.

Methods: The key elements in this solution will be: A) How to transfer the file and metadata. B) How to ensure the security of data transfer. Uploading or downloading data from a web server is easy and this can also be achieved by multiple methods, such as FTP, openSSH, mail server, etc. We use an HTTP server side script because it is easy to use to perform some other related tasks, e.g. operating on the database. The security can be ensured by using SSL/TLS (see section 4.2.2), the SSL server/client side authentication and log in through the SSL tunnel should be secure and easy to develop.

2.2.3 Mail solution

Assumption: The upstream server has a mail server available, and we can put a database trigger/daemon on the upstream server.

Aim: to update the downstream server's database when there is a relevant update of the upstream server's database.

Solution: given an mail server, we can easily send the metadata in the mail and the MSDS file as a MIME (Multipurpose Internet Mail Extensions) encoded object. The downstream server doesn't need to have their own mail server, it can be located anywhere, as long as it can be reached and the new mail can be downloaded. At the downstream side, extracting data from the mail message and inserting data into the database is also not difficult. When an relevant update occurs, the trigger/daemon on the upstream server will start a program on the local machine to put the corresponding metadata and MSDS file into a mail message and deliver it to the configured email

address in order to reach the downstream server. The downstream server should check the mail server frequently or use a mail arrival notice mechanism. After it receives the mail and finishes the updating task, i.e. updating database, it should send a reply mail message to the upstream server to notify it of the success of update, so that both of them can log this event.

Methods: sending mail via a program is not difficult. Many programming tools and applications provide functions to send and receive mail. When sending an e-mail with and PDF file attachment, the receiver will see a text file (the mail content) and the PDF file in his mailbox. This is using the multipart mail format. However, for MIME encoded file, when you attach a file to the mail, the mail application will convert it into a single message body, i.e. the mail content. So when you send this mail, the receiver will only receive a single message body, which contains both the metadata mail content and the attachment. The receiver can then extract each of these using the MIMI encoding information.

Appendix B is an example of a mail message with attachment (in this case a HTML file). The first seven lines are the mail header. The Content-type: multipart/mixed; boundary=”396d983d6b89a” line shows that the mail contains more than one part. It also defines the boundary, thus new parts in the mail will be divided by “- -“and the boundary code. There will be also a “- -“following the last boundary, indicates there is no additional part after it.

2.2.4 Socket programming solution

Assumption: The user is willing to install a daemon which listens to a configured port and to open the corresponding port in their firewall’s configuration.

Aim: to update the downstream server’s database when there is a relevant update on the upstream server’s database.

Solution: Socket programming is more flexible than the previous methods, since we can transfer the data need freely via the connection setup between the upstream and downstream servers. An advantage is that we only need to install an integrated program on each side, and the connection will be continuous. It can be data driven, i.e. based upon a trigger in the database or periodically checking for an update to start the data transfer process. Once an update is available, the daemon on the upstream server will send a connection request to the daemon on the downstream server. Then they setup up the connection, perform the authentication, and then transfer the corresponding data. If the transfer finishes successfully, they log this event on both sides, otherwise they repeat the transfer. The disadvantage of this method is that we need to have a program which can execute on all the different types of computer being used and on each of the operating systems be used. Thus leading to much higher support costs.

Methods: if this solution is selected, there needs to be a host name or IP address and port number in the database, as well as an open port information in both firewalls. Most programming languages provide functions to perform socket operations.

2.3. Comparison of SysM & SysD solutions

Below is a comparison of SysM and SysD solutions we have discussed in the early section, from the table we can see clearly the advantages and disadvantages of the three solutions. Based on the comparison, we will choose a method to use for this project in next chapter.

Items	HTTP web application	Mail solution	Socket programming
Apache server	Yes	No	No
Mail Server	No	Yes	No
Daemon on server side	No	No	Yes
Daemon on client side	No	Yes	Yes
Browser independent	Yes	Yes	Yes
Operating system independent	Yes	Yes	Yes
Client Database Independent*	No	No	No
Delay	Detectable	Undetectable	Detectable
Alert if update fails	Yes	Yes	Yes
Extra Firewall configuration**	No	No	Yes
Security level	High	Medium	High
An application layer solution	Yes	Yes	No
Extra Database Structure	No	No	Yes
Error Risk	Medium	Low	Medium
Developing Period	Medium	Medium	Long

Table 2 comparison of SysD solutions

***Client database independent indicates whether the client side daemon is database independent, because the downstream user may use a database other then Mysql. In the project we assume that they all use Mysql, if not maybe we need to develop a connection tool for the purpose**

****to let our daemon connect to the internet, the downstream user must configure their firewall and give this information to the MSDS service administrator, so it can insert the IP address and port number into the database**

2.4. Emergency issues

There can be also some emergency issues, e.g. what if a customer didn't pay for our service but an accident happens? What if the court finds out that we had the corresponding MSDS? These issues lie outside the technical part of this thesis project, but do need to be considered as part of the business solution which is offered to customers.

Chapter 3: Evaluation

3.1 Selecting the method to be used

In this section we will evaluate the solutions for SysM and SysD discussed above to choose the best solution. Mainly we focus on the following areas.

1. Connection orientated: connection orientated transfers are important, since we want to ensure the prompt receipt of MSDS files and metadata.

The HTTP web application solution can be seen as connection oriented. It keeps the connection open out the whole process, i.e. until a successful event is logged or an error report sent. However, this connection orientation is not very complete, because when the upstream server sends a request to the downstream server, there can be an error when the downstream user tries to connect to it. In another word, the upstream server couldn't detect that an error occurred on the downstream user's connection attempt. However, the upstream server will be aware that it notified the downstream server and has not yet gotten a response. This has little difference from the socket based approach, where again the upstream server can initiate a connection to the downstream server's socket, but it can't know why it does not answer. However, the cause of the failure will be logged in the downstream server's own system. Therefore the HTTP web application solution can be seen as mostly connection oriented.

The mail solution is not connection oriented, since the upstream server doesn't know when the mail is sent. Of course the downstream user is able to send back a reply when it receives the object, but in general the mail solution is asynchrony, which means that each party doesn't know all the details of the state of the data transfer. Therefore the mail solution is not connection oriented.

The socket programming solution is completely connection oriented. The upstream server controls through the whole process and if any error is detected the connection will be lost and an error message should be generated. Therefore the socket programming is connection oriented.

2. Retry: retries are a major issue in this project. Too many unsuccessful attempts can waste system resources as well as generate many error entities.

Errors can be detected as soon as it happens in both the HTTP web application and socket programming approach, thus this minimizing the number of reties. In comparison, in the mail solution it is not so easy to detect and handle the errors. However we have decades of experience with mail delivery systems and most offer extensive logging and audit trails.

3. Security issues: for the HTTP web application solution, since the URL is passed through the SSL/TLS tunnel following mutual authentication—what is requested and received can not be interpreted or altered by third parties. Therefore it can be regarded as secure.

Although the SSL/TLS can also be used in the mail solution, S/MIME is designed

to provide cryptographic security services for email applications, e.g. authentication, message integrity and non-repudiation of origin. However, there are some slight obstacles while deploying it, for example, S/MIME is tailored for end-to-end security. Encryption will not only encrypt the messages, but also malware, thus if your mail is scanned for malware anywhere but at the end points, such as the company's gateway, encryption will defeat the detector and successfully deliver the malware. To learn more about S/MIME and mail security, please refer to [24]

The SSL/TLS structure can also be applied to the socket programming solution, however opening a new socket for incoming traffic requires configuration of the firewall, so this will cause extra work for the users. Of course if the end user's system were to open an outgoing socket to the MSDS service's servers then the work is on the MSDS service and not the end users.

4. Daemons: daemons will consume system resources, therefore fewer daemons are preferred. As a data driven system, HTTP web application won't use any daemon beyond the Apache server (if we can put a trigger in the upstream server's database). Although most mail user agents already have a built-in function for checking received mail, often configurable to check every on or more minutes, but we still need a daemon to extract the metadata from the message body and update the database. Therefore the mail solution needs at least one daemon, however automatic mail processing daemons exist and are quite widely used, see for example, procmail [25]. The socket programming requires at least two daemons, one on each side.

5. Delay: connection oriented solutions have less delay because the system can control the whole process once it is triggered. The mail solution may have greater delay as mail sometimes arrives late (as it depends upon the load on the mail servers which are involved). However, even the socket programming and web server approaches can face added delay due to other tasks running on the same computer, other transfers which are being made, etc.

After considering all of these advantages and disadvantages, I decided to choose the HTTP web application approach as the solution for this project. Not only because of the advantages discussed above, but also because that Antivenena itself already have a MSDS generation system written in PHP, so it will be much easier to integrate them.

3.2 Related work

3.2.1 SUP—Software Update Protocol

SUP [26] is a program for updating files from another machine. It consists of two parts, a file server and the client program. When the client needs to update a file, the client runs *sup* to start the updating process, or the user can run *sup* as a daemon, so that every day it checks for updates and then updates the files to the latest version.

A most powerful feature of SUP is that it can support multiple releases on the

server. Therefore users can specify a file version that they wish to download.

3.2.2 File Distribution Manager

File Distribution Manager [27] is a system used to distribute files via internet or intranet at a specified time or interval. It uses FTP or SMTP to transfer files to the registered clients. It has the following key features:

- File distribution via FTP or SMTP
- Both files and recipients can be grouped, and linked – thus enabling a file or group of files to easily be distributed to a group of recipients
- You can distribute new or only modified files
- Built-in time scheduler for automated operation or it can be invoked manually
- ZIP-archives can be used for file compression and password protected can be enabled
- Logging and monitoring
- Database driven (Borland Databases Engine)

These features enable the system to transfer the files properly, correctly and reliably to the registered clients. The main disadvantage of this system is it uses Borland Database Engine as the database, which Borland does no longer support. To read more details about disadvantages of using the Borland Database Engine, please refer to [28].

3.2.3 Adaptive File Distribution Protocol

Adaptive File Distribution Protocol (AFDP) [28] is a protocol designed for efficient distribution of large files to many hosts on a LAN. In AFDP any machine receiving data is called a ***subscriber*** while a machine sending data is called a ***publisher***. This protocol is built on top of UDP, and was developed to automatically determine the best transport mode, depending upon the number of subscribers, their capabilities, and support from the networks. This protocol works is similar to the program *rcp*, but capitalizing on the broadcast and multicast facilities of UDP. And AFDF also allows the user to control the trade-off between speed, host load, and network load, therefore it is quite flexible and easy to use.

3.2.4 LiteCopy

LiteCopy [29] is designed to offer a secure, reliable and easily implemented solution for file transfer over the internet. It is written in Java and use http(s) protocol to transfer the files. Therefore it has the ability to run on UNIX, Linux, Windows, and other platforms that have a Java Virtual Machine installed. Only the http port (most 80)

or https port (most 443) needs to be available for this program. Additionally, one of the advantages of the system is that no client side installation or configuration is needed, because they use a web browser with a Java plug-in to initiate data transfers. LiteCopy also uses SSL/TLS to secure the connection between clients and the server. It can also set the client privileges on the server side, so that the client can not abuse its operation access to the server.

3.2.5 WebDAV

Web-based Distributed Authoring and Versioning (WebDAV) [30] is a protocol aimed to make the WWW a *readable* and *writable* medium. It is used to provide functionalities for creating, changing, and moving documents on a remote server. The most important aim of its design is to enable multiple users to cooperate on the distributing of a document. A major advantage of this protocol is that today most operating systems provide support for it, hence it is wide spread. WebDAV is standardized in RFC 2518 [31].

Chapter 4: Implementation Approach

4.1 Selected Method

The main aim of this project is to provide a swift and secure MSDS updating service for downstream users. It should be easy to develop and maintain. This service should be based upon widely used technology, rather than special purpose code. While there are many technologies (as presented in chapter 2), that could be used to achieve these goals (as presented in section 1.2.2). Considering each of these approaches, I realized that this project should be a push method for convenience and practical reasons.

After all the consideration, I choose an implementation based upon the HTTP web application solution discussed in section 2.2.2.

This solution is naturally data driven since updates occur only when new updates are actually ready. Thus at the upstream server we must learn that there is an update ready, and then we need to propagate this information to the downstream servers. Learning that there is an update is easy for upstream server, as this can be based upon a database trigger or daemon—we assume that the database and upstream server are running on the same machine or are both within a trusted domain, hence the process of notifying the upstream server that there is an update should not be difficult. Therefore the main problem is to notify the downstream user that a relevant update is ready. Following the authentication of both parties, the data transfer process should start.

We are all familiar with the web Browser-Server architecture, by using a server side script we can start a program on another computer (the server) based upon a request from a browser (a client). Therefore the server side script can be used to notify the downstream server that there is an update ready. Thus the notice will be in such an order, the upstream server—where the update has occurred—sends a request to designed server side script on the downstream user saying that here is an update, hence it should start the authentication and data transfer process. Here the updated data successfully triggers the upstream server (based upon the trigger or daemon) to send a request to the downstream server, just as if it were a user browsing the downstream server's web, this invokes a server side script at the downstream server, which in turn causes it to request the update from the upstream server. Note that the main reason to have the downstream server make a request of the upstream server for the update is that by doing so each party only has to open their firewall for normal HTTP traffic.

After the downstream server has received the notification of an update, it connects to the upstream server and starts the authentication process. Because the downstream server connected to the upstream server's secure web interface, all of the traffic is sent through the SSL/TLS tunnel, here we assume that mutual authentication

is done by both parties. Thus it will ensure the security of the data transfer. After mutual authentication the actual data transfer occurs.

The method can be viewed as one web server notifying another that there is an update and telling it that the second web server can fetch the update from the first web server. The core of this method is that both servers act as a client when communication with the other.

After examining the existing tools, I decide to use the technologies described in section 4.2 for this implementation. The main advantages of these tools are: first, they can implement this method properly, that is when there are updates to the database of either the manufacture or the MSDS service, the program triggers the downstream server(s) to update its(their) database(s); second it is easy to develop and should be easy to maintain; third, it will be sufficiently secure-- while avoiding extra configuration, particularly of any firewalls; finally, all the software installed on the manufacturers' or downstream users' servers are well-known and widely used, so it will be much easier for them to trust the software.

4.2: Technical Details of the Proposed Solution

4.2.1 Linux, Apache, MySQL, and either Perl, Python, or PHP (LAMP)

LAMP – Linux [3], Apache [5], MySQL [6], and either Perl, Python, or PHP [7], will be used as the main technical platform of this project. It is easy to understand from the name that it is a combination of the four technologies used within the platform. LAMP provides an open source web platform for development and deployment of high performance web applications [2].

The Apache HTTP server [5] is an open source web server. It has been adopted by many websites. By May 1999 Apache served 57% of all websites, and by February 2006, this had increased to 68% [4]. It was first developed by Rob McCool and other developers. When it was first developed, it was the only available open source alternative to the Netscape web server (now known as Sun Java System Web Server). Currently its main competitor is Microsoft's Internet Information Services (IIS).

Apache offers many advantages to the users, developers, and web administrators. It is customizable so that you can build a server that is made to measure. It implements the latest web protocols and is extensible. The Apache server and application programming interface (API) source code are open. It is efficient, as lots of effort has been put into optimizing Apache's code for performance. As a result, it runs faster and consumes less system resources than many other servers.

Since it is open source software, when bugs are found, they can be quickly communicated and rapidly fixed, which makes it quite reliable. Its configuration file is in ASCII, having a simple format, so it can be edited using any text editor.

MySQL [6] is the most popular open source database in the world. It is a multithreaded, multi-user SQL Database Management System, and can be mostly used for free for non-commercial uses. However, if it is used commercially in this application Antivenena will need to get a license, since they would be susing it for commercial purpose. MySQL AB holds the copyright for most of the core code, and sells support and service contracts, as well as develops and maintains the system. It is also easy to configure and administer. As it is open source, there exist many third party tools to provide extensions to it.

PHP [7] is a powerful server side scripting language for creating dynamic and interactive websites. It is free and widely used. It can be embedded directly into HTML, and it is very similar to C in style.

PHP generally runs on the web servers, but command line scripting and client side applications are also part of the three primary uses of PHP. It can be deployed on any web server and most operating system platforms.

LAMP has been used to design websites for quite a long time, and has proven to be highly functional, portable, and browser independent. It can be installed on all the existing systems and supports all current browsers. There are some packages, such as AppServ [21], which integrates Apache, PHP, MySql, PHPmyAdmin and MyODBC, so you can just simply download it and install on you server.

4.2.2 SSL/TLS – Secure Sockets Layer and Transport Layer Security

We use SSL/TLS to authenticate the parties and to protect the subsequent communication. SSL/TLS was developed by Netscape Communication and it provides a so called security service layer, which is between the application layer and the TCP layer. SSL v3 after slight adjustments was adopted as an IETF standard under the name of TLS. It is commonly used in HTTP to form HTTPS, which is widely used for applications such as e-commerce and on-line banking systems.

SSL/TLS provides methods to prevent eavesdropping, tampering, and message forgery for applications communicating over a network. It mainly involves three basic phases:

1. Peer negotiation for algorithm support
2. Public key encryption based key exchange and certificate based authentication
3. Symmetric cipher –based traffic encryption

During the first phase, the client and server negotiation which cryptographic algorithm they will use. Currently the following choices are supported:

- For public key cryptography: RSA, Diffie-Hellman, DSA, or Fortezza.
- For symmetric ciphers: RC2, RC4, IDEA, DES, Triple DES, AES, or Camellia.
- For one-way hash functions: MD2, MD4, MD5, or SHA

SSL/TLS mostly uses RSA public key [8] cryptography to do the authentication.

Public key encryption is a technique that uses a pair of asymmetric keys for encryption and decryption. Every pair of the keys consists of a public key and a private key. The public key is public and distributed widely. The private key is kept as a secret. Data that is encrypted with the public key can only be decrypted by the private key, which is kept as a secret by the owner, so it is supposed to be secure enough unless the key is stolen. Conversely, the data encrypted with the private key can be decrypted by the public key, which ensures that only the client can read the encrypted data. Such asymmetry is the main property which makes public key cryptography so useful.

In the SSL/TLS protocol, an object, called a Certificate was invented to prevent a man-in-the-middle attack. A certificate has the following content:

- The certificate issuer's name
- The entity for which it is being issued
- The public key of the subject
- Some time stamps

The certificate is signed by the issuer's private key, and everyone is supposed to know this user's public key, which means the issuer needs to be trusted by all parties. By using certificate technology, everybody can examine the sender's certificate to see whether it's been forged -- as long as they trust the certificate authority and the certificate has not been revoked.

To read more about SSL/TLS please see [9].

4.2.3 PEAR – PHP Extension and Application Repository

PEAR is a framework and distribution system for PHP components, just like APIs for C++. And I used some functions it provides in the proposed system. Its purpose is to provide:

- a structured library of open source code for PHP users
- a system for code distribution and package maintenance
- a standard style for code written in PHP
- the PHP Extension Community Library (PECL)
- a web site mailing list and download mirrors to support the PHP/PEAR community

Nowadays PEAR provides lots functions in many areas for Authentication, Database, File formats, networking, etc. To learn more about PEAR, please visit [10].

4.3 Scenario

A manufacture has just updated a MSDS, and they want to notify the MSDS service's server of this update so it can get it in order that this service can update its database. Thus the MSDS service is to download the MSDS file and insert the metadata into its database. Therefore the manufacture sends a request to the MSDS service to achieve the following goals: 1. Notify it that an MSDS has been updated. 2.

Send the URL that will be used to fetch this updated MSDS. 3. Sign the request so that the information can not be altered without detection and to prove that the request comes from the manufacture (which should be regarded as the MSDS service's upstream customer).

When the MSDS service receives the request and confirms that is from a valid subscriber, it will make a request to the specified URL—which was contained in the manufacture's request, thus the manufacture can recognize this is the expected request and it will permit the requestor to download the relevant MSDS file and corresponding metadata. After downloaded the file, the manufacture will update its database to record that the MSDS service has received the update.

When the MSDS service has successfully received the MSDS and the associated metadata, it will carefully evaluate its quality, and then included it into its database. After that, it will notify the relevant customers of this updated MSDS, by initiating the distribution process in turn to its downstream customers.

The distribution process is nearly the same, after bidirectional authentication, the customers will be able to download the MSDS service's authorized MSDS file. Note that both sides will update their database to indicate that they have received (respectively delivered) the updated MSDS file.

4.4 Design and Implementation

First we need to configure the Apache server to support PHP and bidirectional SSL authentication. To configure the Apache server to support PHP, we need to add two lines in the HTTPd.conf file, they are "AddType application/x-HTTPd-php .php" and "LoadModule php4_module phpModuleFile".

Since the MSDS service needs to identify its customers and the customer also needs to identify the MSDS service, bidirectional authentication is required. To configure the Apache server to support bidirectional SSL authentication, we need four certificates: a private key certificate, a public key certificate, a certificate authority (CA), and a CAchain. Then we also have to configure the following properties in the configuration file.

- SSLCertificateFile
- SSLCertificateKeyFile
- SSLCACertificateFile
- SSLVerifyClient
- SSLVeryfyDepth

To learn more about how to configure these values, please refer to [22].

After completing this configuration, the system now works. In the following text, I will use *upstream server* and *downstream server* to indicate the data provider and receiver (respectively).

Step 1: After the upstream has generated the new version MSDS file and inserted the corresponding metadata into the database. The upstream server notifies the downstream server about this update. This will be done via sending a request to the

down stream server. This request will be something like this HTTPs://downstreamserver/send.php?url=HTTPs:// upstreamserver/get.php&fileid=YYYYYYY&filesize=99. Note sending this request is the notification! The fileid variable is used to identify the corresponding file and the filesize variable is used to check for errors, i.e. in some cases the download process may suffer some errors and the file may be incomplete. The file size provides an easy way to check for a complete transfer. Note that bidirectional authentication will occur.

Step 2: The downstream server downloads the indicated file and metadata. When the downstream server received the request, it learned the following information though the client certificate and request variables. 1. Who is offering the MSDS update. 2. Where to get the file and metadata. 3. The name of the file. 4. The size of the file. Thus the downstream server can send a request of form HTTPs://upstreamserver/get.php?fileid=YYYYYYY to get the corresponding file and metadata.

Step 3: Both servers update their database and log this event. After the download process, the downstream user checks the integrity of the file and updates its database. It then sends another request to notify the upstream server that everything is finished or failed. This request has the form: https://upstreamserver/successful.php&fileid=YYYYYYY or https://upstreamserver/failure.php?fileid=YYYYYYY. At this point the upstream server will update its own database to indicate the downstream server has the newest version if the transfer was successful, otherwise if explicit failure is noted or a timeout occurs, the upstream server will log this fact and repeat the attempt up to a predefined number of times; if each of these fails, then the operator is notified to take appropriate action. And both sides will log this event. If any error happens during this process, an email should be sent to the administrator to report the error.

The whole process is shown in figure 3.

Figure 3 Work process

4.5 Measurement

Measurements are used to evaluate this application. In addition a suitable test suited need to be developed to thoroughly test should be made before putting this application to actual usage. This section gives the results of these tests as well as describing the test environment used. In this test, the transfer speed, CPU usage, and memory usage are measured. The test environment consists of:

- Upstream server: Intel Centrino 1.6G/512M, Windows XP. The server should be run under Linux, but as most of my developing job is done in a Vmware Virtual Machine Linux under WinXP (a Vmware Virtual Machine can totally give you a Linux environment under Windows, so you don't need to switch between two systems very often), I setup the test environment under Windows to avoid unnecessary system resource consuming and get a better result. It can also test that whether the system is operation system independent.
- Downstream server: AMD duron 933/384M, Windows XP.

The two servers are in the same network and the network speed is 100M, they are given the IP address 192.168.0.1 and 192.168.0.2 (respectively).

Process Explorer [23] is used to gather the system resource information.

The SSL/TLS information exchange is captured by Ethereal [32] (see Appendix D, this is captured in another test, so the ip addresses are not the same). In this test, only the server authentication is enabled.

In the test, 100 files are transferred between the two servers. These files totally are 45MB. In fact, they are the same 450 Kbytes files with different names 729.pdf .. 828.pdf. Using such file names makes the test much easier, since it is easy to write a script to loop over a set of files names.

The transferring time for all of these files are 22 seconds (see Appendix C) and the highest CPU usage of the Apache server (the upstream server) is 18.04% (see figure 4, the highest point of the green line is 18.04%). In an earlier test, 500 files (totaling almost 300MB) are transferred; the CPU usage was always below 40% although its takes longer time to finish all of these transfers. Thus the bottleneck is not the processor or software, and it is simply limited by the file size and network speed.

In figure 4, the red line indicates cpu usage in kernel-mode whereas green line is the sum of kernel-mode and user-mode execution. And in figure 5, the blue line indicates the total I/O traffic, which is the sum of all process I/O reads and writes, and the pink line shows the write traffic.

Comparing figure 4 and figure 5, we can see the highest cpu usage occurred when there are writing operations, which mostly happens when the program needs to write the temp file, and it becomes regularly after the mass write period, which peaks every time the apache server reads files from the disk. From figure 6, we can see the memory usage of the apache server is quite steady around 30MB.

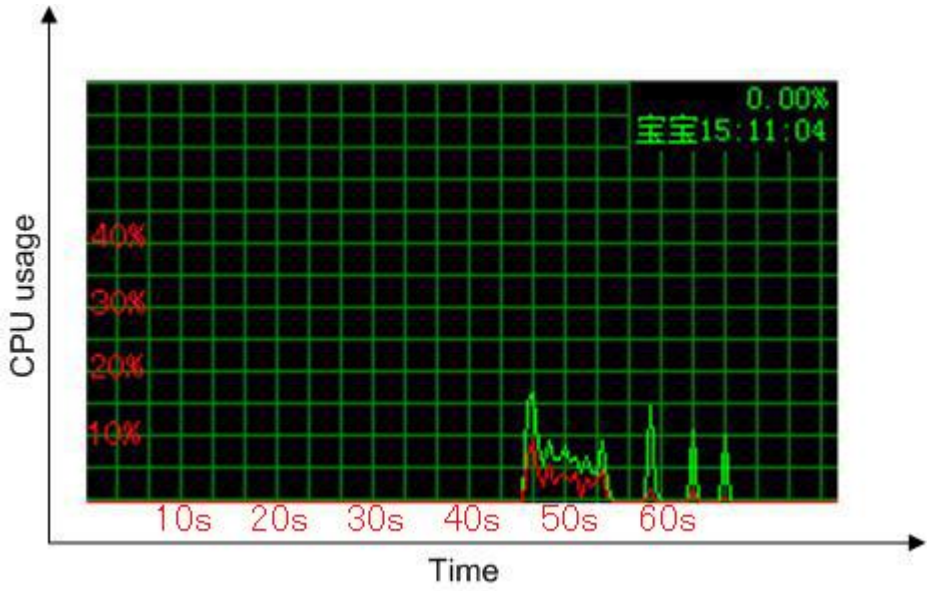

Figure 4: CPU usage

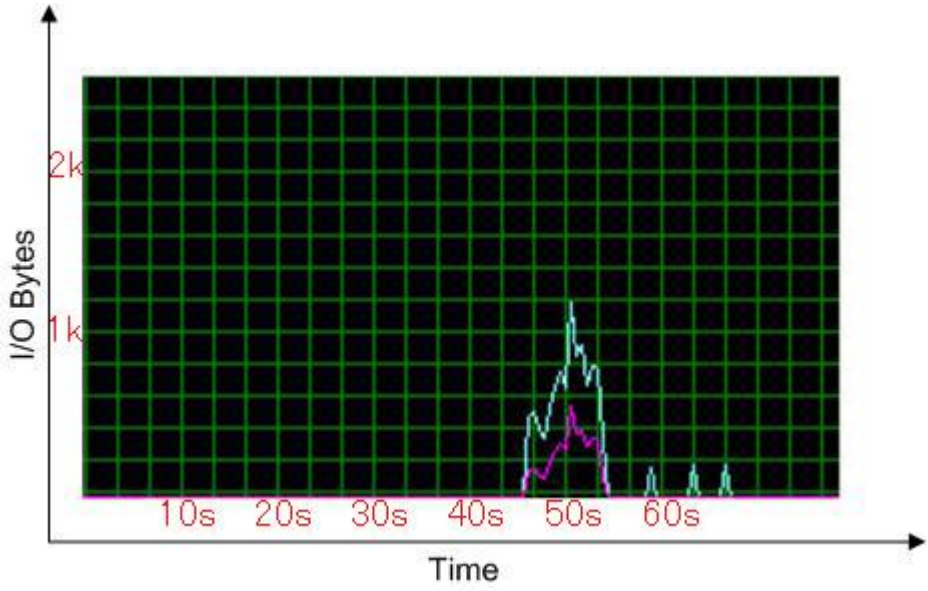

Figure 5: I/O bytes

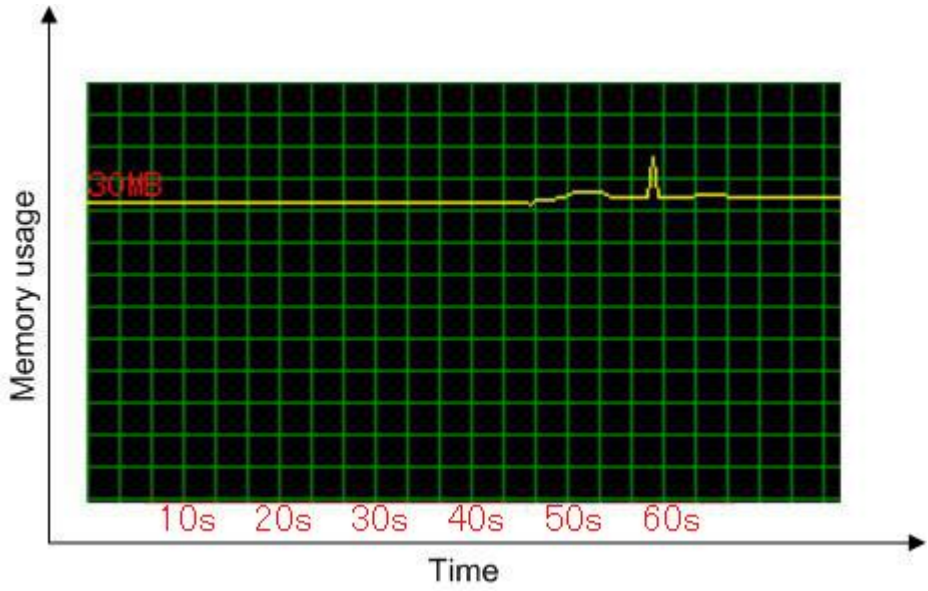

Figure 6: Memory usage

We can see from the log file (Appendix C) that for some cases the file transfer may finish somewhat later, but it causes no severe problems. The average time of the transfer and database update for a 0.45M file is less then 1 second in the test environment. And more then 2M file can be transfer in 1 second, which means nearly 7200M/hour. So it is fast enough to meet Antivenena's requirement, which is transferring nearly 1200M within one hour.

4.6 SSL/TLS Stream Analysis

From the captured SSL/TLS stream we can see how the SSL/TSL works clearly. As shown in Figure 7.

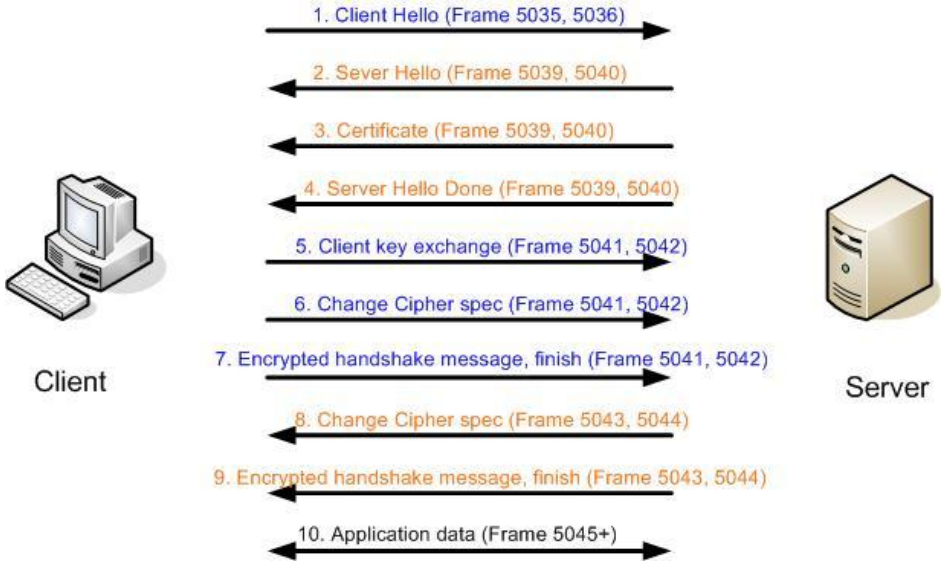

Figure 7 SSL/TLS data stream

1. The first step in the process is the client sending a Client Hello message to the server (Frame 5035, 5036). This hello message should contain the SSL version and the cipher suites the client can talk. The client sends its maximum key length details at this time.

2. The server replies the client hello message with one of its own in which it nominates the version of SSL and the ciphers and key length to be used in the conversation, chosen from the choice offered in the client hello.

3. The server sends its certificate to the client.

4. The server sends a server done message noting it has concluded the initial part of the setup sequence. Step 2, 3, 4 are done in two frames—Frame 5039 and 5040.

5. The client generates a symmetric key and encrypts it using the server's public key (included in the certificate). It then sends this message to the server.

6. The client sends a cipher spec message telling the server all future communication should be with the new key.

7. The client sends a finish message using the new key to determine if the server is able to decrypt the message and the negotiation was successful. Step 5, 6, 7 are done in two frames—Frame 5041 and 5042.

8. The server sends a Change Cipher Spec message telling the client that all future communications will be encrypted.

9. The server sends its own encrypted finish message using the key. If the client can read this message then the negotiation is successfully completed. Step 8, 9 are done in two frames—Frame 5043 and 5044.

10. Transfer application data with the agreed encryption method.

4.7 Comparison with existing file distribution systems

In this section we will compare this system with the existing file distribution systems introduced in section 3.2.

1. Comparing with SUP, the proposed system uses commonly used http(s) port, so that no need to open a new port at the firewall. The system is data driven using a push method, while SUP is a pull method which means the client will connect to the server and pull the updates, thus unnecessary connections may take place. The SUP supports multiple releases while this system doesn’t, since it is not useful, as the clients should have only one release—the latest version.
2. Comparing with File Distribution Manager, the proposed system uses http(s) while File Distribution Manager uses FTP or SMTP, which are also commonly used. The advantage of this system is it uses the mysql database, which is thought to have a better future then BDE that File Distribution

Manager uses, which is no longer support by Borland--its creator. A weakness of the proposed system is that compression files and password protection is not supported, maybe in the future, this function can be added to save transfer time and increase security if some customers require. But the PDF files are already LZW compressed so there is likely to be little more compression gain to be had. And reducing redundancy generally makes the system more prone to error.

3. Comparing with AFDP, the proposed system works in the application layer and it focuses on the proper transfer result, which ensures the clients all over the internet get the updates, and AFDP is a protocol built on top of UDP and focuses on the best transfer mode in a LAN. Therefore they focus on different area.
4. Comparing with LiteCopy, the proposed system doesn't require the action of any program or plug-in in the clients, however it does require some php files to be copied to the web server directory. Thus there is no risk to the client computer. Whereas, the JAVA plug-in used in LiteCopy sometimes may cause some trouble for the browser. Additionally, for people who use third party plug-in deleting tools, it is possible to delete the LiteCopy plug-in by mistake.
5. WebDAV focuses on making the WWW a readable and writable medium, thus it has a different purpose then the proposed system. However, WebDAV enables multiple users to cooperate on the document while the proposed system is limited to distributing an edited document to all relevant clients. Thus a subset of WebDAV's capabilities may be used to achiever the purpose of the proposed system of distributing files, but the proposed system also needs to transfer the metadata and insert them into the client databases, that is what WebDAV misses.

4.8 Limitations and future work

A limitation in this method will be that we need to have a signed certificate from a trusted authority, such as the EU Chemical Agency. Additionally, each downstream user needs to install a client certificate.

The future work for this project should investigate an easier way to distribute MSDS file to many down stream users together. This might be done by sending email to a mailing list, as this might save some system resources since some unnecessary file preparation is saved, that is because in the current project, each file need to transfer is given a random temperate name for each user to avoid abuse downloading. However, in current project, a connection oriented solution was taken to ensure the prompt delivery of the MSDS file and metadata.

Chapter 5: Conclusion

The aim of the project is to introduce an efficient and secure MSDS file and corresponding metadata transfer system, so that the downstream users of Antivenena can update the MSDS files to newer versions timely.

The overall conclusion of this thesis project is that the method and implementing tools can achieve our goals. The system has the following features.

1. It uses HTTP(s) protocol. So if will just use the port 80 or 443.
2. It is data driven push method, so the client don't need to run any program manually or update in the configured period.
3. It uses popular web building tools---- LAMP and some php pages, so the clients just need to have web server and tend to be trust it.
4. As it runs as some web pages, so no special install and configuration needed as long as the web server support php and SSL, and no information will be written to system register.
5. It uses the commonly used SSL/TLS to secure the information change, so it can be seen secure enough.

Therefore it can be seen as a kind of combination or tailor of the existing file distribution tools to meet our requirement, like little installation, commonly used tools, enough secure, and data driven method. This system meets the technical goals we set in section 1.2.2. Generally, it doesn't modify too much on the existing database structure, and the added table or fields properly present the data needed in the file distribution process. The SysM and SysD were developed in the project, and they use SSL/TLS to ensure the security of data transfer. Both SysM and SysD are independent from operating system and all the important events are logged and can be traced. And it is able to transfer more then 7200M files within one hour, that can meet the requirement, which is transfer nearly 1200M files per hour.

The main requirement of the Antivenena is to transfer the MSDS files and corresponding metadata safely and swiftly, due to the measurement done in section 4.5, we can see the proposed system can provide an acceptable service which meets the requirement.

References

[1] REACH, [www], HTTP://ec.europa.eu/environment/chemicals/reach/reach_intro.htm, last access, 01-Dec-2006

[2] Dale Dougherty, *LAMP: The Open Source Web Platform*, [www] HTTP://www.onlamp.com/pub/a/onlamp/2001/01/25/lamp.html, Last access 06-Dec-2006.

[3] Linux Online, [www] HTTP://www.linux.org, Last access on 07-Dec-2006

[4] Wikipeida, Apache HTTP Server, [www] HTTP://en.wikipedia.org/wiki/Apache_HTTP_Server, last access 02-Dec-2006

[5] Apache Software Foundation, [www] HTTP://www.apache.org, last access 02-Dec-2006

[6] MySQL AB, [www] HTTP://www.mysql.com, last access 02-Dec-2006

[7] PHP: Hypertext Preprocessor, [www] HTTP://www.php.net, last access 02-Dec-2006

[8] RSA Security, [www] HTTPs://www.rsasecurity.com/ , last access 02-Dec-2006

[9] T. Dierks, C. Allen, *The TLS Protocol*, [www] HTTP://www.ietf.org/rfc/rfc2246.txt, last access 02-Dec-2006

[10] PEAR, [www] HTTP://pear.php.net, last access 02-Dec-2006

[11] US standard from the US department of labor: HTTP://www.osha.gov/pls/oshaweb/owalink. query_links?src_doc_type=STANDARDS&src_unique_file=1910_1200&src_anchor_name=1910.1200(g) (US style), last access 11-Dec-2006

[12] The recent EU REACH, [www] HTTP://register.consilium.europa.eu/PDF/en/06/st07/st07524.en06.PDF, last access 11-Dec-2006

[13] Q and A on the new Chemicals policy REACH (EU Style) HTTP://europa.eu.int/rapid/pressReleasesAction.do?reference=MEMO/03/213&format=HTML&aged=0&language=EN&guiLanguage=en, last access 11-Dec-2006

[14] Explanation of MSDS: HTTP://www.des.umd.edu/os/rtk/msds/explanation.html, last access 11-Dec-2006

[15] MSDS FAQ: HTTP://www.ilpi.com/msds/, last access 11-Dec-2006

[16] XML format MSDS: HTTPs://www.denix.osd.mil/denix/Public/Library/MSDS/HMIS/hmis.html, last access 12-Dec-2006

[17] Web-based Distributed Authoring and Versioning: HTTP://www.webdav.org/, last access 11-Dec-2006

[18] PDF MSDS example: HTTP://www.chemdat.info/documents/sds/emd/bel/nl/8150/815015.PDF, last access 11-Dec-2006
(EU Style)
Html MSDS example: HTTP://www.ilpi.com/msds/vx.html (US Style), last access 11-Dec-2006

[19] Conseq homepage: HTTP://www.conseq.co.uk/, last access 11-Dec-2006

[20] The wercs homepage: HTTP://www.thewercs.com/solutions/tools.asp?lang=en, last access 14-Dec-2006

[21] Tyler Thompson, *AppServ: Web Service Package*, [www] HTTP://www.pcmech.com/show/opensource/794/, last access 16-Dec-2006

[22] mod_ssl reference, [www] HTTP://www.modssl.org/docs/2.1/ssl_reference.html, last access 18-Jan-2007

[23] Process Explorer, [www] http://www.microsoft.com/technet/sysinternals/utilities/ProcessExplorer.mspx, last access 09-Feb-2007

[24] S/MIME wikipedia, [www], http://en.wikipedia.org/wiki/S/MIME, last access 09-Feb-2007

[25] Procmail, wikipedia, [www], http://en.wikipedia.org/wiki/Procmail, last access 24-Feb-2007

[26] SUP,[www], http://www.rocketaware.com/man/man1/sup.1.htm, last access, 24-Feb-2007

[27] File Distribution Manager, [www], http://www.duodata.de/fdm/index.htm, last access, 24-Feb-2007

[28] Why not use BDE, [www], http://www.componentace.com/info/articles/why_not_use_bde.php, last access, 24-Feb-2007

[29] LiteCopy, SoftLink Ltd., Petah Tikva, Israel, [www] , http://www.litecopy.com/litecopy_non_techie.html last access, 08-Apr-2007

[30] WebDAV, [www], http://en.wikipedia.org/wiki/WebDAV, last access, 08-Mar-2007

[31] Y.Goland, E. Whitehead, A. Faizi, S. Carter, and D. Jensen, "HTTP Extensions for Distributed Authoring – WEBDAV", RFC 2518, [www], http://www.ietf.org/rfc/rfc2518.txt, last access, 08-Apr-2007

[32] Ethereal, [www], http://www.ethereal.com/, last access, 08-Mar-2007

Abbreviations

CA	Certificate Authority
DBMS	Database Management System
IETF	The Internet Engineering Task Force
IIS	Internet Information Services
LAMP	Linux, Apache, MySQL, and either Perl, Python, or PHP
MSDS	Material Safety Data Sheet
PEAR	PHP Extension and Application Repository
PECL	PHP Extension Community Library
REACH	Registration, Evaluation and Authorization of Chemicals
S/MIME	Secure / Multipurpose Internet Mail Extensions
SSL/TLS	Secure Sockets Layer and Transport Layer Security
SysM	System we are developing between Manufacture and MSDS service
SysD	System we are developing between MSDS service and downstream users

Appendix A MSDS example

(from: HTTP://physchem.ox.ac.uk/MSDS/EU/eugenol.html)

Safety (MSDS) data for eugenol

General

Synonyms: 2-methoxy-4-(2-propenyl)phenol
Molecular formula: $CH_2CH_2CH_2C_6H_3(OCH_3)OH$
CAS No: 97-53-0
EC No:

Physical data

Appearance: colourless liquid with a strong odour of cloves
Melting point: -9 C
Boiling point: 253 C
Vapour density:
Vapour pressure:
Density (g cm^{-3}): 1.06
Flash point: 110 C (closed cup)
Explosion limits:
Autoignition temperature:
Water solubility: negligible

Stability

Stable. Combustible. Incompatible with strong oxidizing agents.

Toxicology

May act as a skin or eye irritant.

Toxicity data

(The meaning of any abbreviations which appear in this section is given here.)
ORL-RAT LD50 2680 mg kg^{-1}
ORL-MUS LD50 3000 mg kg^{-1}
IPR-RAT LDLO 800 mg kg^{-1}
IPR-MUS LD50 500 mg kg^{-1}

Risk phrases

(The meaning of any risk phrases which appear in this section is given here.)
R36 R38.

Transport information

Personal protection

Minimise contact.

Safety phrases

(The meaning of any safety phrases which appear in this section is given here.)

[Return to Physical & Theoretical Chemistry Lab. Safety home page.]

This information was last updated on September 26, 2003. We have tried to make it as accurate and useful as possible, but can take no responsibility for its use, misuse, or accuracy. We have not verified this information, and cannot guarantee that it is up-to-date.

Appendix B Typical Mail with Attachment

Return-Path:
Date: Mon, 22 May 2000 19:17:29 +0000
From: Someone
To: Person
Message-id: <83729KI93LI9214@example.com>
Content-type: multipart/mixed; boundary="396d983d6b89a"
Subject: Here@#s the subject
--396d983d6b89a
Content-type: text/plain; charset=iso-8859-1
Content-transfer-encoding: 8bit

This is the body of the email.

--396d983d6b89a
Content-type: text/html; name=attachment.html
Content-disposition: inline; filename=attachment.html
Content-transfer-encoding: 8bit

This is the attached HTML file

--396d983d6b89a--

Appendix C Test Log File

request.php_730.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
request.php_731.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
request.php_732.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
request.php_734.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
request.php_733.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
request.php_735.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
fileready.php_down1_730.pdf_8.1_108403_Friday 16th of February 2007 03:33:29 PM
request.php_736.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
request.php_737.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
fileready.php_down1_731.pdf_8.1_108403_Friday 16th of February 2007 03:33:29 PM
request.php_738.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
fileready.php_down1_732.pdf_8.1_108403_Friday 16th of February 2007 03:33:29 PM
fileready.php_down1_734.pdf_8.1_108403_Friday 16th of February 2007 03:33:29 PM
request.php_739.pdf_8.1_Friday 16th of February 2007 03:33:29 PM
fileready.php_down1_733.pdf_8.1_108403_Friday 16th of February 2007 03:33:30 PM
request.php_740.pdf_8.1_Friday 16th of February 2007 03:33:30 PM
fileready.php_down1_735.pdf_8.1_108403_Friday 16th of February 2007 03:33:30 PM
request.php_741.pdf_8.1_Friday 16th of February 2007 03:33:30 PM
request.php_742.pdf_8.1_Friday 16th of February 2007 03:33:30 PM
fileready.php_down1_736.pdf_8.1_108403_Friday 16th of February 2007 03:33:30 PM
fileready.php_down1_737.pdf_8.1_108403_Friday 16th of February 2007 03:33:30 PM
fileready.php_down1_738.pdf_8.1_108403_Friday 16th of February 2007 03:33:30 PM
request.php_743.pdf_8.1_Friday 16th of February 2007 03:33:30 PM
request.php_745.pdf_8.1_Friday 16th of February 2007 03:33:30 PM
request.php_744.pdf_8.1_Friday 16th of February 2007 03:33:30 PM
Success_down1____730.pdf___8.1_Friday 16th of February 2007 03:33:30 PM
Success_down1____732.pdf___8.1_Friday 16th of February 2007 03:33:31 PM
request.php_746.pdf_8.1_Friday 16th of February 2007 03:33:31 PM
fileready.php_down1_741.pdf_8.1_108403_Friday 16th of February 2007 03:33:31 PM
fileready.php_down1_739.pdf_8.1_108403_Friday 16th of February 2007 03:33:31 PM
Success_down1____731.pdf___8.1_Friday 16th of February 2007 03:33:31 PM
fileready.php_down1_740.pdf_8.1_108403_Friday 16th of February 2007 03:33:31 PM
Success_down1____733.pdf___8.1_Friday 16th of February 2007 03:33:31 PM
request.php_747.pdf_8.1_Friday 16th of February 2007 03:33:31 PM
request.php_748.pdf_8.1_Friday 16th of February 2007 03:33:31 PM
request.php_750.pdf_8.1_Friday 16th of February 2007 03:33:31 PM
request.php_749.pdf_8.1_Friday 16th of February 2007 03:33:31 PM
request.php_751.pdf_8.1_Friday 16th of February 2007 03:33:31 PM
fileready.php_down1_742.pdf_8.1_108403_Friday 16th of February 2007 03:33:32 PM
Success_down1____734.pdf___8.1_Friday 16th of February 2007 03:33:32 PM

request.php_752.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_754.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_753.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
fileready.php_down1_743.pdf_8.1_108403_Friday 16th of February 2007 03:33:32 PM
fileready.php_down1_745.pdf_8.1_108403_Friday 16th of February 2007 03:33:32 PM
request.php_755.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_756.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_758.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_757.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_759.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_760.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
Success_down1____735.pdf___8.1_Friday 16th of February 2007 03:33:32 PM
fileready.php_down1_744.pdf_8.1_108403_Friday 16th of February 2007 03:33:32 PM
request.php_761.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_762.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_763.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_764.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
Success_down1____738.pdf___8.1_Friday 16th of February 2007 03:33:32 PM
request.php_765.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_766.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
request.php_767.pdf_8.1_Friday 16th of February 2007 03:33:32 PM
Success_down1____737.pdf___8.1_Friday 16th of February 2007 03:33:32 PM
Success_down1____736.pdf___8.1_Friday 16th of February 2007 03:33:32 PM
request.php_768.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_769.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_771.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_770.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_774.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_773.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_772.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_776.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
fileready.php_down1_746.pdf_8.1_108403_Friday 16th of February 2007 03:33:33 PM
request.php_775.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_778.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_777.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_779.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_781.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_780.pdf_8.1_Friday 16th of February 2007 03:33:33 PM
request.php_782.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_783.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
Success_down1____741.pdf___8.1_Friday 16th of February 2007 03:33:34 PM
request.php_784.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_785.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_786.pdf_8.1_Friday 16th of February 2007 03:33:34 PM

request.php_787.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
Success_down1____739.pdf___8.1_Friday 16th of February 2007 03:33:34 PM
request.php_789.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_788.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_790.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_792.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_791.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_794.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_793.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
fileready.php_down1_748.pdf_8.1_108403_Friday 16th of February 2007 03:33:34 PM
fileready.php_down1_747.pdf_8.1_108403_Friday 16th of February 2007 03:33:34 PM
Success_down1____740.pdf___8.1_Friday 16th of February 2007 03:33:34 PM
request.php_795.pdf_8.1_Friday 16th of February 2007 03:33:34 PM
request.php_796.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_797.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_798.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_799.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
fileready.php_down1_750.pdf_8.1_108403_Friday 16th of February 2007 03:33:35 PM
fileready.php_down1_749.pdf_8.1_108403_Friday 16th of February 2007 03:33:35 PM
request.php_800.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_802.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_801.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
fileready.php_down1_751.pdf_8.1_108403_Friday 16th of February 2007 03:33:35 PM
request.php_803.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_805.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_806.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_804.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_807.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_809.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
fileready.php_down1_752.pdf_8.1_108403_Friday 16th of February 2007 03:33:35 PM
request.php_808.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_810.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_811.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
request.php_812.pdf_8.1_Friday 16th of February 2007 03:33:35 PM
fileready.php_down1_754.pdf_8.1_108403_Friday 16th of February 2007 03:33:35 PM
request.php_813.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
request.php_814.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
fileready.php_down1_753.pdf_8.1_108403_Friday 16th of February 2007 03:33:36 PM
request.php_815.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
request.php_816.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
Success_down1____742.pdf___8.1_Friday 16th of February 2007 03:33:36 PM
request.php_817.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
request.php_818.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
fileready.php_down1_756.pdf_8.1_108403_Friday 16th of February 2007 03:33:36 PM

fileready.php_down1_755.pdf_8.1_108403_Friday 16th of February 2007 03:33:36 PM
request.php_819.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
request.php_820.pdf_8.1_Friday 16th of February 2007 03:33:36 PM
request.php_821.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_758.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
request.php_823.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
request.php_822.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
Success_down1____743.pdf___8.1_Friday 16th of February 2007 03:33:37 PM
request.php_824.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
request.php_825.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
request.php_826.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
Success_down1____745.pdf___8.1_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_757.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
request.php_827.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_760.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_759.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
request.php_828.pdf_8.1_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_761.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_762.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_764.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_763.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_765.pdf_8.1_108403_Friday 16th of February 2007 03:33:37 PM
fileready.php_down1_766.pdf_8.1_108403_Friday 16th of February 2007 03:33:38 PM
Success_down1____744.pdf___8.1_Friday 16th of February 2007 03:33:38 PM
fileready.php_down1_767.pdf_8.1_108403_Friday 16th of February 2007 03:33:38 PM
fileready.php_down1_768.pdf_8.1_108403_Friday 16th of February 2007 03:33:38 PM
fileready.php_down1_769.pdf_8.1_108403_Friday 16th of February 2007 03:33:38 PM
fileready.php_down1_771.pdf_8.1_108403_Friday 16th of February 2007 03:33:38 PM
fileready.php_down1_774.pdf_8.1_108403_Friday 16th of February 2007 03:33:38 PM
fileready.php_down1_770.pdf_8.1_108403_Friday 16th of February 2007 03:33:38 PM
fileready.php_down1_773.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_772.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_776.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_775.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_778.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_779.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_777.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_780.pdf_8.1_108403_Friday 16th of February 2007 03:33:39 PM
fileready.php_down1_781.pdf_8.1_108403_Friday 16th of February 2007 03:33:40 PM
fileready.php_down1_782.pdf_8.1_108403_Friday 16th of February 2007 03:33:40 PM
fileready.php_down1_783.pdf_8.1_108403_Friday 16th of February 2007 03:33:40 PM
fileready.php_down1_784.pdf_8.1_108403_Friday 16th of February 2007 03:33:40 PM
fileready.php_down1_785.pdf_8.1_108403_Friday 16th of February 2007 03:33:40 PM
fileready.php_down1_786.pdf_8.1_108403_Friday 16th of February 2007 03:33:40 PM

fileready.php_down1_787.pdf_8.1_108403_Friday 16th of February 2007 03:33:40 PM
fileready.php_down1_789.pdf_8.1_108403_Friday 16th of February 2007 03:33:41 PM
Success_down1____746.pdf___8.1_Friday 16th of February 2007 03:33:41 PM
fileready.php_down1_788.pdf_8.1_108403_Friday 16th of February 2007 03:33:41 PM
fileready.php_down1_792.pdf_8.1_108403_Friday 16th of February 2007 03:33:41 PM
fileready.php_down1_790.pdf_8.1_108403_Friday 16th of February 2007 03:33:41 PM
fileready.php_down1_791.pdf_8.1_108403_Friday 16th of February 2007 03:33:41 PM
fileready.php_down1_793.pdf_8.1_108403_Friday 16th of February 2007 03:33:41 PM
fileready.php_down1_794.pdf_8.1_108403_Friday 16th of February 2007 03:33:41 PM
fileready.php_down1_795.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_799.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_796.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_800.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_797.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_798.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_802.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_804.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_801.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_803.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_806.pdf_8.1_108403_Friday 16th of February 2007 03:33:42 PM
fileready.php_down1_805.pdf_8.1_108403_Friday 16th of February 2007 03:33:43 PM
fileready.php_down1_807.pdf_8.1_108403_Friday 16th of February 2007 03:33:43 PM
fileready.php_down1_809.pdf_8.1_108403_Friday 16th of February 2007 03:33:43 PM
fileready.php_down1_810.pdf_8.1_108403_Friday 16th of February 2007 03:33:43 PM
fileready.php_down1_808.pdf_8.1_108403_Friday 16th of February 2007 03:33:43 PM
fileready.php_down1_812.pdf_8.1_108403_Friday 16th of February 2007 03:33:43 PM
fileready.php_down1_811.pdf_8.1_108403_Friday 16th of February 2007 03:33:43 PM
Success_down1____747.pdf___8.1_Friday 16th of February 2007 03:33:43 PM
Success_down1____748.pdf___8.1_Friday 16th of February 2007 03:33:43 PM
fileready.php_down1_813.pdf_8.1_108403_Friday 16th of February 2007 03:33:44 PM
fileready.php_down1_815.pdf_8.1_108403_Friday 16th of February 2007 03:33:44 PM
Success_down1____750.pdf___8.1_Friday 16th of February 2007 03:33:44 PM
fileready.php_down1_814.pdf_8.1_108403_Friday 16th of February 2007 03:33:44 PM
fileready.php_down1_816.pdf_8.1_108403_Friday 16th of February 2007 03:33:44 PM
fileready.php_down1_817.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_819.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_818.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_820.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
Success_down1____749.pdf___8.1_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_821.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_823.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_822.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_825.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
Success_down1____751.pdf___8.1_Friday 16th of February 2007 03:33:45 PM

fileready.php_down1_824.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_826.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_828.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
Success_down1____752.pdf___8.1_Friday 16th of February 2007 03:33:45 PM
fileready.php_down1_827.pdf_8.1_108403_Friday 16th of February 2007 03:33:45 PM
Success_down1____754.pdf___8.1_Friday 16th of February 2007 03:33:46 PM
Success_down1____753.pdf___8.1_Friday 16th of February 2007 03:33:46 PM
Success_down1____756.pdf___8.1_Friday 16th of February 2007 03:33:46 PM
Success_down1____755.pdf___8.1_Friday 16th of February 2007 03:33:47 PM
Success_down1____758.pdf___8.1_Friday 16th of February 2007 03:33:47 PM
Success_down1____760.pdf___8.1_Friday 16th of February 2007 03:33:47 PM
Success_down1____757.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____759.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____761.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____762.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____764.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____765.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____763.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____766.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____767.pdf___8.1_Friday 16th of February 2007 03:33:48 PM
Success_down1____769.pdf___8.1_Friday 16th of February 2007 03:33:49 PM
Success_down1____768.pdf___8.1_Friday 16th of February 2007 03:33:49 PM
Success_down1____771.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____770.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____774.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____773.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____772.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____775.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____778.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____776.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____779.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____780.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____781.pdf___8.1_Friday 16th of February 2007 03:33:50 PM
Success_down1____777.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____782.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____783.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____784.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____786.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____785.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____787.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____789.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____792.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____788.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____790.pdf___8.1_Friday 16th of February 2007 03:33:51 PM

Success_down1____791.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____793.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____794.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____795.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____799.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____796.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____798.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____800.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____802.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____797.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____801.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____805.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____806.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____803.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____804.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____807.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____809.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____810.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____811.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____815.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____812.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____808.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____813.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____816.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____814.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____817.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____819.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____818.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____820.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____821.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____823.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____822.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____825.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____828.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____824.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____826.pdf___8.1_Friday 16th of February 2007 03:33:51 PM
Success_down1____827.pdf___8.1_Friday 16th of February 2007 03:33:51 PM

Appendix D SSL Transferred Data

No. Time Source Destination Protocol Info
5035 222.241650 192.168.0.108 192.168.0.133 SSLv2 Client Hello

Frame 5035 (157 bytes on wire, 157 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.360269000
Time delta from previous packet: 0.019644000 seconds
Time since reference or first frame: 222.241650000 seconds
Frame Number: 5035
Packet Length: 157 bytes
Capture Length: 157 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.1 (00:19:5b:da:49:48)
Destination: 192.168.0.1 (00:19:5b:da:49:48)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4108 (Normal)
Duration: 44
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.108 (00:19:5b:81:41:17)
Destination address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 3786
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 1, Ack: 1, Len: 63
Secure Socket Layer

No. Time Source Destination Protocol Info
5036 222.242210 192.168.0.108 192.168.0.133 SSLv2 [TCP Retransmission] Client Hello

Frame 5036 (117 bytes on wire, 117 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.360829000
Time delta from previous packet: 0.000560000 seconds
Time since reference or first frame: 222.242210000 seconds
Frame Number: 5036

Packet Length: 117 bytes
Capture Length: 117 bytes
Protocols in frame: eth:ip:tcp:ssl
Ethernet II, Src: 192.168.0.108 (00:19:5b:81:41:17), Dst: 192.168.0.133 (00:0e:35:b5:54:55)
Destination: 192.168.0.133 (00:0e:35:b5:54:55)
Source: 192.168.0.108 (00:19:5b:81:41:17)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 1, Ack: 1, Len: 63
Secure Socket Layer

No. Time Source Destination Protocol Info
5039 222.262211 192.168.0.133 192.168.0.108 SSLv3 Server Hello, Certificate, Server Key Exchange, Server Hello Done

Frame 5039 (1136 bytes on wire, 1136 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.380830000
Time delta from previous packet: 0.007235000 seconds
Time since reference or first frame: 222.262211000 seconds
Frame Number: 5039
Packet Length: 1136 bytes
Capture Length: 1136 bytes
Protocols in frame: eth:ip:tcp:ssl:pkcs-1:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:pkcs-1:pkcs-1
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1, Ack: 64, Len: 1082
Secure Socket Layer

No. Time Source Destination Protocol Info
5040 222.263944 192.168.0.133 192.168.0.108 SSLv3 [TCP Retransmission] Server Hello, Certificate, Server Key Exchange, Server Hello Done

Frame 5040 (1176 bytes on wire, 1176 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.382563000
Time delta from previous packet: 0.001733000 seconds
Time since reference or first frame: 222.263944000 seconds

Frame Number: 5040
Packet Length: 1176 bytes
Capture Length: 1176 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl:pkcs-1:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:pkcs-1:pkcs-1
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4208 (Normal)
Duration: 117
Destination address: 192.168.0.108 (00:19:5b:81:41:17)
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 390
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1, Ack: 64, Len: 1082
Secure Socket Layer

No.	Time	Source	Destination	Protocol	Info
5041	222.280621	192.168.0.108	192.168.0.133	SSLv3	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message

Frame 5041 (308 bytes on wire, 308 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.399240000
Time delta from previous packet: 0.016677000 seconds
Time since reference or first frame: 222.280621000 seconds
Frame Number: 5041
Packet Length: 308 bytes
Capture Length: 308 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.1 (00:19:5b:da:49:48)
Destination: 192.168.0.1 (00:19:5b:da:49:48)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)

Frame Control: 0x4108 (Normal)
Duration: 44
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.108 (00:19:5b:81:41:17)
Destination address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 3789
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 64, Ack: 1083, Len: 214
Secure Socket Layer

No. Time Source Destination Protocol Info
5042 222.281239 192.168.0.108 192.168.0.133 SSLv3 [TCP Retransmission] Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message

Frame 5042 (268 bytes on wire, 268 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.399858000
Time delta from previous packet: 0.000618000 seconds
Time since reference or first frame: 222.281239000 seconds
Frame Number: 5042
Packet Length: 268 bytes
Capture Length: 268 bytes
Protocols in frame: eth:ip:tcp:ssl
Ethernet II, Src: 192.168.0.108 (00:19:5b:81:41:17), Dst: 192.168.0.133 (00:0e:35:b5:54:55)
Destination: 192.168.0.133 (00:0e:35:b5:54:55)
Source: 192.168.0.108 (00:19:5b:81:41:17)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 64, Ack: 1083, Len: 214
Secure Socket Layer

No. Time Source Destination Protocol Info
5043 222.294440 192.168.0.133 192.168.0.108 SSLv3 Change Cipher Spec, Encrypted Handshake Message

Frame 5043 (129 bytes on wire, 129 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.413059000
Time delta from previous packet: 0.013201000 seconds
Time since reference or first frame: 222.294440000 seconds
Frame Number: 5043

Packet Length: 129 bytes
Capture Length: 129 bytes
Protocols in frame: eth:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1083, Ack: 278, Len: 75
Secure Socket Layer

No.	Time	Source	Destination	Protocol	Info
5044	222.295378	192.168.0.133	192.168.0.108	SSLv3	[TCP Retransmission] Change Cipher Spec, Encrypted Handshake Message

Frame 5044 (169 bytes on wire, 169 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.413997000
Time delta from previous packet: 0.000938000 seconds
Time since reference or first frame: 222.295378000 seconds
Frame Number: 5044
Packet Length: 169 bytes
Capture Length: 169 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4208 (Normal)
Duration: 117
Destination address: 192.168.0.108 (00:19:5b:81:41:17)
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 392
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1083, Ack: 278, Len: 75
Secure Socket Layer

No. Time Source Destination Protocol Info
5045 222.298891 192.168.0.108 192.168.0.133 SSLv3 Application Data

Frame 5045 (899 bytes on wire, 899 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.417510000
Time delta from previous packet: 0.003513000 seconds
Time since reference or first frame: 222.298891000 seconds
Frame Number: 5045
Packet Length: 899 bytes
Capture Length: 899 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.1 (00:19:5b:da:49:48)
Destination: 192.168.0.1 (00:19:5b:da:49:48)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4108 (Normal)
Duration: 44
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.108 (00:19:5b:81:41:17)
Destination address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 3790
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 278, Ack: 1158, Len: 805
Secure Socket Layer

No. Time Source Destination Protocol Info
5046 222.299818 192.168.0.108 192.168.0.133 SSLv3 [TCP Retransmission] Application Data

Frame 5046 (859 bytes on wire, 859 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.418437000
Time delta from previous packet: 0.000927000 seconds
Time since reference or first frame: 222.299818000 seconds
Frame Number: 5046
Packet Length: 859 bytes
Capture Length: 859 bytes
Protocols in frame: eth:ip:tcp:ssl

Ethernet II, Src: 192.168.0.108 (00:19:5b:81:41:17), Dst: 192.168.0.133 (00:0e:35:b5:54:55)
Destination: 192.168.0.133 (00:0e:35:b5:54:55)
Source: 192.168.0.108 (00:19:5b:81:41:17)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 278, Ack: 1158, Len: 805
Secure Socket Layer

No.	Time	Source	Destination	Protocol	Info
5047	222.321228	192.168.0.108	192.168.0.133	SSLv3	Application Data, Application Data, [Unreassembled Packet]

Frame 5047 (1554 bytes on wire, 1554 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.439847000
Time delta from previous packet: 0.021410000 seconds
Time since reference or first frame: 222.321228000 seconds
Frame Number: 5047
Packet Length: 1554 bytes
Capture Length: 1554 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.1 (00:19:5b:da:49:48)
Destination: 192.168.0.1 (00:19:5b:da:49:48)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4108 (Normal)
Duration: 44
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.108 (00:19:5b:81:41:17)
Destination address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 3791
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 1083, Ack: 1158, Len: 1460
Secure Socket Layer
[Unreassembled Packet: SSL]

No.	Time	Source	Destination	Protocol	Info
5048	222.322495	192.168.0.108	192.168.0.133	SSLv3	[TCP

Retransmission] Application Data, Application Data, [Unreassembled Packet]

Frame 5048 (1514 bytes on wire, 1514 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.441114000
Time delta from previous packet: 0.001267000 seconds
Time since reference or first frame: 222.322495000 seconds
Frame Number: 5048
Packet Length: 1514 bytes
Capture Length: 1514 bytes
Protocols in frame: eth:ip:tcp:ssl
Ethernet II, Src: 192.168.0.108 (00:19:5b:81:41:17), Dst: 192.168.0.133 (00:0e:35:b5:54:55)
Destination: 192.168.0.133 (00:0e:35:b5:54:55)
Source: 192.168.0.108 (00:19:5b:81:41:17)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 1083, Ack: 1158, Len: 1460
Secure Socket Layer
[Unreassembled Packet: SSL]

No. Time Source Destination Protocol Info
5049 222.322924 192.168.0.133 192.168.0.108 TCP https > 1085 [ACK] Seq=1158 Ack=2543 Win=14978 Len=0

Frame 5049 (54 bytes on wire, 54 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.441543000
Time delta from previous packet: 0.000429000 seconds
Time since reference or first frame: 222.322924000 seconds
Frame Number: 5049
Packet Length: 54 bytes
Capture Length: 54 bytes
Protocols in frame: eth:ip:tcp
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1158, Ack: 2543, Len: 0

No. Time Source Destination Protocol Info
5050 222.323178 192.168.0.133 192.168.0.108 TCP [TCP Window Update] https > 1085 [ACK] Seq=1158 Ack=2543 Win=17520 Len=0

Frame 5050 (54 bytes on wire, 54 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.441797000
Time delta from previous packet: 0.000254000 seconds
Time since reference or first frame: 222.323178000 seconds
Frame Number: 5050
Packet Length: 54 bytes
Capture Length: 54 bytes
Protocols in frame: eth:ip:tcp
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1158, Ack: 2543, Len: 0

No.	Time	Source	Destination	Protocol	Info
5051	222.323771	192.168.0.133	192.168.0.108	TCP	[TCP Window Update] https > 1085 [ACK] Seq=1158 Ack=2543 Win=14978 Len=0

Frame 5051 (94 bytes on wire, 94 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.442390000
Time delta from previous packet: 0.000593000 seconds
Time since reference or first frame: 222.323771000 seconds
Frame Number: 5051
Packet Length: 94 bytes
Capture Length: 94 bytes
Protocols in frame: eth:wlan:llc:ip:tcp
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4208 (Normal)
Duration: 117
Destination address: 192.168.0.108 (00:19:5b:81:41:17)
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 395
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)

Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1158, Ack: 2543, Len: 0

No. Time Source Destination Protocol Info
5052 222.324199 192.168.0.108 192.168.0.133 SSLv3
Continuation Data, [Unreassembled Packet]

Frame 5052 (1554 bytes on wire, 1554 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.442818000
Time delta from previous packet: 0.000428000 seconds
Time since reference or first frame: 222.324199000 seconds
Frame Number: 5052
Packet Length: 1554 bytes
Capture Length: 1554 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.1 (00:19:5b:da:49:48)
Destination: 192.168.0.1 (00:19:5b:da:49:48)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4108 (Normal)
Duration: 44
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.108 (00:19:5b:81:41:17)
Destination address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 3792
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 2543, Ack: 1158, Len: 1460
Secure Socket Layer
[Unreassembled Packet: SSL]

No. Time Source Destination Protocol Info
5053 222.324306 192.168.0.133 192.168.0.108 TCP [TCP Window Update] https > 1085 [ACK] Seq=1158 Ack=2543 Win=17520 Len=0

Frame 5053 (94 bytes on wire, 94 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.442925000
Time delta from previous packet: 0.000107000 seconds
Time since reference or first frame: 222.324306000 seconds

Frame Number: 5053
Packet Length: 94 bytes
Capture Length: 94 bytes
Protocols in frame: eth:wlan:llc:ip:tcp
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4208 (Normal)
Duration: 117
Destination address: 192.168.0.108 (00:19:5b:81:41:17)
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 396
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1158, Ack: 2543, Len: 0

No. Time Source Destination Protocol Info
5054 222.325554 192.168.0.108 192.168.0.133 SSLv3 [TCP Fast Retransmission] Continuation Data, [Unreassembled Packet]

Frame 5054 (1514 bytes on wire, 1514 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.444173000
Time delta from previous packet: 0.001248000 seconds
Time since reference or first frame: 222.325554000 seconds
Frame Number: 5054
Packet Length: 1514 bytes
Capture Length: 1514 bytes
Protocols in frame: eth:ip:tcp:ssl
Ethernet II, Src: 192.168.0.108 (00:19:5b:81:41:17), Dst: 192.168.0.133 (00:0e:35:b5:54:55)
Destination: 192.168.0.133 (00:0e:35:b5:54:55)
Source: 192.168.0.108 (00:19:5b:81:41:17)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 2543, Ack: 1158, Len: 1460
Secure Socket Layer
[Unreassembled Packet: SSL]

No. Time Source Destination Protocol Info
5055 222.325947 192.168.0.108 192.168.0.133 SSLv3
Continuation Data, [Unreassembled Packet]

Frame 5055 (1445 bytes on wire, 1445 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.444566000
Time delta from previous packet: 0.000393000 seconds
Time since reference or first frame: 222.325947000 seconds
Frame Number: 5055
Packet Length: 1445 bytes
Capture Length: 1445 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.1 (00:19:5b:da:49:48)
Destination: 192.168.0.1 (00:19:5b:da:49:48)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4108 (Normal)
Duration: 44
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.108 (00:19:5b:81:41:17)
Destination address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 3793
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 4003, Ack: 1158, Len: 1351
Secure Socket Layer
[Unreassembled Packet: SSL]

No. Time Source Destination Protocol Info
5056 222.326141 192.168.0.133 192.168.0.108 TCP https > 1085 [ACK] Seq=1158 Ack=4003 Win=17520 Len=0

Frame 5056 (54 bytes on wire, 54 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.444760000
Time delta from previous packet: 0.000194000 seconds
Time since reference or first frame: 222.326141000 seconds
Frame Number: 5056
Packet Length: 54 bytes

Capture Length: 54 bytes
Protocols in frame: eth:ip:tcp
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.108 (00:19:5b:81:41:17)
Destination: 192.168.0.108 (00:19:5b:81:41:17)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.133 (192.168.0.133), Dst: 192.168.0.108 (192.168.0.108)
Transmission Control Protocol, Src Port: https (443), Dst Port: 1085 (1085), Seq: 1158, Ack: 4003, Len: 0

No.	Time	Source	Destination	Protocol	Info
5057	222.326755	192.168.0.108	192.168.0.133	SSLv3	Application Data

Frame 5057 (147 bytes on wire, 147 bytes captured)
Arrival Time: Mar 30, 2007 12:24:33.445374000
Time delta from previous packet: 0.000614000 seconds
Time since reference or first frame: 222.326755000 seconds
Frame Number: 5057
Packet Length: 147 bytes
Capture Length: 147 bytes
Protocols in frame: eth:wlan:llc:ip:tcp:ssl
Ethernet II, Src: 192.168.0.133 (00:0e:35:b5:54:55), Dst: 192.168.0.1 (00:19:5b:da:49:48)
Destination: 192.168.0.1 (00:19:5b:da:49:48)
Source: 192.168.0.133 (00:0e:35:b5:54:55)
Type: IEEE 802.11 (Centrino promiscuous) (0x2452)
IEEE 802.11
Type/Subtype: Data (32)
Frame Control: 0x4108 (Normal)
Duration: 44
BSS Id: 192.168.0.1 (00:19:5b:da:49:48)
Source address: 192.168.0.108 (00:19:5b:81:41:17)
Destination address: 192.168.0.133 (00:0e:35:b5:54:55)
Fragment number: 0
Sequence number: 3794
WEP parameters
Logical-Link Control
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 5354, Ack: 1158, Len: 53
Secure Socket Layer

No.	Time	Source	Destination	Protocol	Info
5058	222.327237	192.168.0.108	192.168.0.133	SSLv3	[TCP

Retransmission] Continuation Data, [Unreassembled Packet]

Frame 5058 (1405 bytes on wire, 1405 bytes captured)
    Arrival Time: Mar 30, 2007 12:24:33.445856000
    Time delta from previous packet: 0.000482000 seconds
    Time since reference or first frame: 222.327237000 seconds
    Frame Number: 5058
    Packet Length: 1405 bytes
    Capture Length: 1405 bytes
    Protocols in frame: eth:ip:tcp:ssl
Ethernet II, Src: 192.168.0.108 (00:19:5b:81:41:17), Dst: 192.168.0.133 (00:0e:35:b5:54:55)
    Destination: 192.168.0.133 (00:0e:35:b5:54:55)
    Source: 192.168.0.108 (00:19:5b:81:41:17)
    Type: IP (0x0800)
Internet Protocol, Src: 192.168.0.108 (192.168.0.108), Dst: 192.168.0.133 (192.168.0.133)
Transmission Control Protocol, Src Port: 1085 (1085), Dst Port: https (443), Seq: 4003, Ack: 1158, Len: 1351
Secure Socket Layer
[Unreassembled Packet: SSL]

COS/CCS 2007-18

www.kth.se