Snow Agent System
Pilot Deployment version

Security policy

Revision: 1.0

Authors:
Per Atle Bakkevoll, Johan Gustav Bellika, Lars Ilebrekke, Taridzo Chomutare

Revision history:

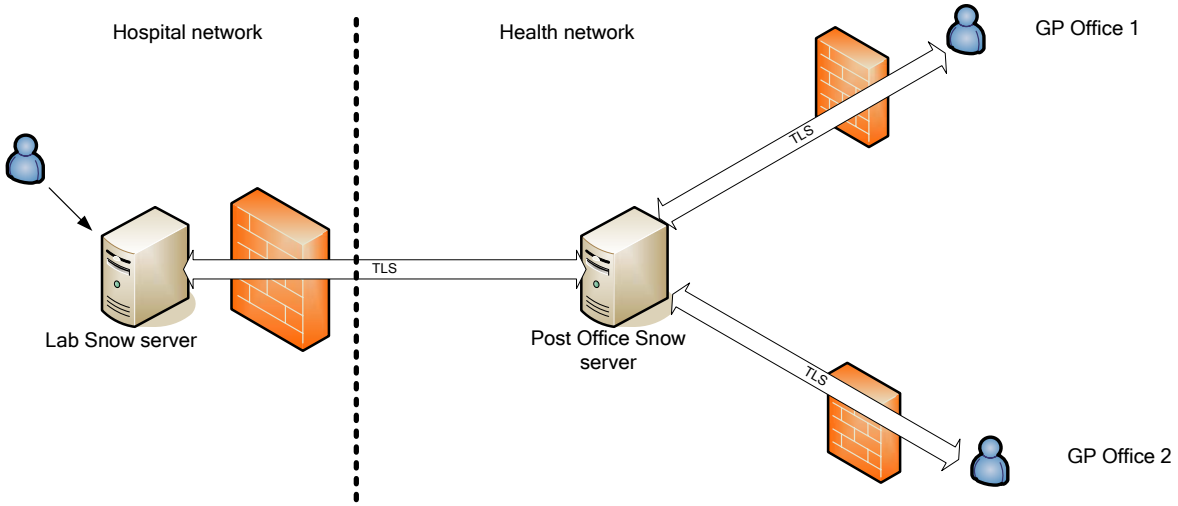
Issue	Details	Who	Date
0.1	Initial version.	L. Ilebrekke	2008.12.08
0.2	Removed the vulnerabilities of PKI and how to reduce the risk.	L. Ilebrekke	2008.12.10
0.3	Added MUC comment and some other clarifications upon comments from Taridzo and Per Atle.	L. Ilebrekke	2008.12.11
0.4	Updated after meeting with the whole team.	L. Ilebrekke	2008.12.17
0.5	Update after comments from Gustav.	L. Ilebrekke	2008.12.22
0.6	Updated end-to-end encryption section and some other minor changes.	L. Ilebrekke	2009.01.09
0.7	Updated with changes made by Gustav (key distribution and authentication between client and server) in a version sent to Helse Nord IKT, January 19th.	L. Ilebrekke	2009.03.11
0.8	Updated after meeting with Gudleif.	L. Ilebrekke	2009.06.08
1.0	Document updated after review by Per Atle and Gustav.	L. Ilebrekke	2009.06.12

Abbreviations:

1 Introduction

This document describes the security policy for the Snow Agent System for the deployment of the pilot version (spring 2009). The system is based on XMPP[1] (Extensible Messaging and Presence Protocol). The XMPP server implementation used is Openfire[2] from Jive software.

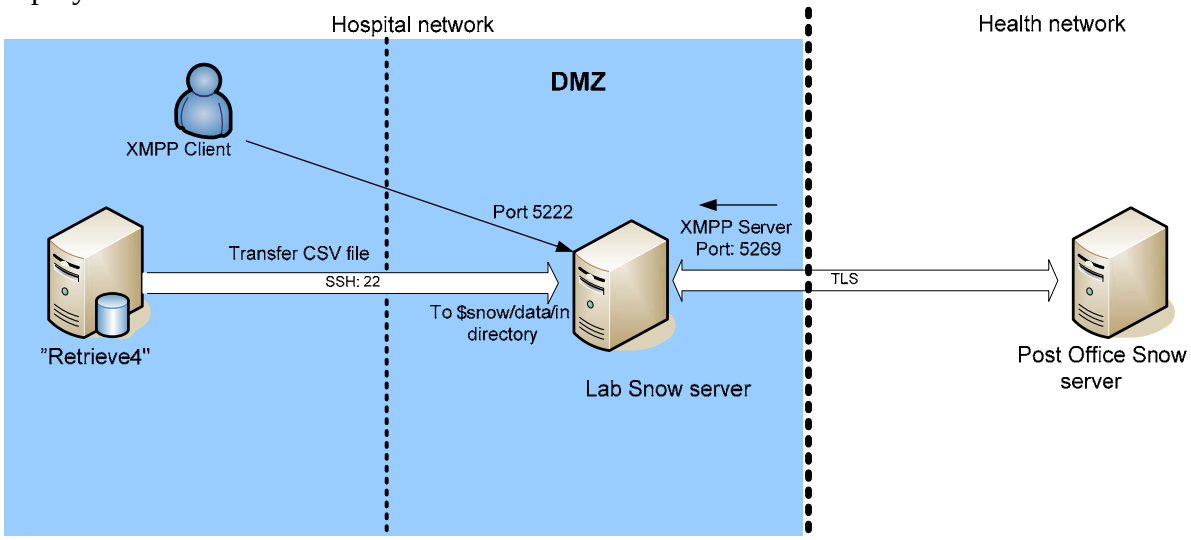
The pilot deployment of the system does not require that GP offices set up Snow servers, but it requires that a Snow server is installed in the hospital's lab. GPs that want to use the system must be added as users to a central server, the Post Office (PO) server, and obtain the necessary certificates. The figure below illustrates the environment.



In general, a Snow XMPP server is necessary on every site where data/statistics is going to be produced. However, in this version it does not apply to the GP offices since they are given the possibility to access lab data without contributing with data produced within their office.

1.1 Deployment view for the hospital's lab

This section provides more details on the hospital's side. The figure below depicts a deployment view.



[1] XMPP web page: http://xmpp.org
[2] Openfire project web page: http://www.igniterealtime.org/projects/openfire/index.jsp

Within the hospital's network there is a demilitarized zone (DMZ) for communication with external entities. Since the lab Snow server needs an XMPP server connection with the PO server in the health net, it must reside in the DMZ. The purpose of the lab Snow server is to retrieve data from the lab database and provide XMPP access to the lab users. Users (XMPP client) within the lab can connect to the server on port 5222. Exported database data (CSV file) are copied to the lab Snow server on an established SSH connection (port 22) from the database server to the lab Snow server. The file is copied to the "$snow/data/in" directory. No connection is made from the lab Snow server to the database server (named Retrieve4) in the lab since information can only be pushed from the inside. The database server is a Windows server running SQL Server 2005.

The patient ID in the lab's database is part of the exported data. The patient ID is not the personal identification number, but it can be used to find right person. To ensure that the exported data can not be traced back to a person, the patient ID is SHA[3] hashed before the export is transferred to the DMZ.

1.2 Server-to-server communication and firewalls

The XMPP standard explicitly states that there has to be two connections between XMPP servers. This means that the firewall in front of the hospital's lab server must be opened up for the PO server. The port to be used for server-to-server communication is port 5269.

1.3 Client-to-server communication and firewalls

The client-to-server communication only requires one connection, which is initiated from the client side. This means that the firewall between the lab and the DMZ does not need any modifications to allow local computers to connect to the Snow server (port 5222) as XMPP clients.

2 Transport Layer Security (TLS)

The server-to-server and the client-to-server communication are done over an established TLS connection. The certificates used must be CA signed and verified upon TLS establishment.

3 Authentication

In the XMPP server-to-server communication and client-to-server communication, SASL EXTERNAL is used. The authentication occurs after TLS has been established. The server is configured to ask the client to provide its certificate and prove that is knows the private key. In server-to-server authentication the other server authenticates with the same domain as in the 'from' address. The client only sends an authenticate message without an identity since it has already proven its identity during TLS, and it only has one XMPP address. That is, the client certificate is for an identity that can be used in all domains, not only for one or a few specific domains.

4 End-to-end message encryption

The XMPP client and XMPP server is able to use end-to-end encryption on the contents of messages, and it is configurable. The XMPP server typically refers to components that receives and sends messages from users and agents. The XMPP client refers to both users and

[3] SHA stands for Secure Hash Algorithm.

agents providing services on the server. The PO server and the lab server can also be configured not to accept messages in clear text. XML encryption is used with the following concepts:

- Generate a symmetric key (AES 256)
- Encrypt the message with the symmetric key.
- For each recipient's public key (RSA):
    - Encrypt the symmetric key with the recipient's public key.
    - Add the EncryptedKey element to the message.

By using symmetric encryption of the message we only need to encrypt the message once, disregarding the number of recipients.

End-to-end message encryption is very hard to support in the multi-user chat (MUC) scenario. The reason is the concept of multi-user and that the chat is served by a server. The XMPP client explicitly states the difference between a private chat with another user and a multi-user chat session.

5 Public Key Infrastructure (PKI)

The keys (public and private) to be generated are based on RSA and x509 certificates.

Private keys are stored in an encrypted key store on the computers hard disk. This is an initial solution and smart cards will be used as soon as it is applicable. The private key must be distributed in a secure way upon creation of a user. A combination of e-mail and cell phone number can be regarded as a transfer mechanism.

Public keys are stored on the XMPP server and can be queried for by the client. There is a distribution protocol between the XMPP servers to make sure that the public keys are updated. The PO server is the master and is the place for revoking certificates. XMPP clients query the local server for the public keys of other users. The clients do not store these public keys to avoid synchronization issues when keys change.

5.1 Private key management

With smart cards these administrative tasks would be quite different and the system could rely on an already established structure and routine. However, this is not an option in phase one of the system, and thus, we need a way to manage the private keys.

The keys are generated centrally, which means that the private key must be sent to the GP in a secure way. The following describes the routine for providing the GP with the private key:

- The GP downloads the client from a server and provides required user information. The user information includes the Health Personnel Number (HPNR) which uniquely identifies the user and a cell phone number. This is done though a Web page made available by Helse-Nord IKT.
- The Snow administrator generates the keys and creates a certificate based on the information provided.
- The Snow administrator encrypts the private key with a randomly selected password.
- The Snow administrator checks the health net address register and sends the encrypted private key in an e-mail to the GP.

- The Snow administrator sends the password to the cell phone number registered to become a user in the system[4].
- The GP starts the client software, runs private key import and provides the password received on his cell phone.
- The GP also selects a personal key or password which is used to encrypt the private key on the client machine.
- When the GP wants to use the private key, he must provide the personal key to decrypt the private key.

The routine described assumes that the information provided in the health register can be trusted not to contain manipulated data.

6 Client functionality

The basic client functionality is to search for statistics (limited search criteria), confirm or send outbreak alerts and communicate with other GPs (chatting). The latter functionality needs specific mechanisms to make the user verify that it does not contain sensitive information. The Snow client will open a dialog window when a new chat connection is initiated or replied to, to make the user confirm that the chat will not contain sensitive or classified information.

The PO server can use a mechanism to verify that the client used is a correct Snow type of client. The mechanism is not intended to be very strong. This means that if a person within the system wants to use another XMPP client and make it look like a Snow client it will be possible. The purpose by providing a client is to make it easier to use and more restricted than any generic XMPP client is.

7 Integration with lab server

All statistical data extracted in this version of the system comes from the hospital's lab. The other parts of this document describe the mechanisms for identity, privacy and confidentiality in the Snow system. This chapter focuses on integrity and consistency of data as well as reliability. The following concerns are addressed and discussed:

1. How to make sure that no sensitive data is extracted by Snow?
    - The search function provides parameters from the client and not database queries.
    - The data access layer used for retrieving data from the lab's database contains prepared SQL statements in which no personal information is extracted from the database.
    - When prepared statements are used correctly it is not possible to use SQL injection type of attacks.
2. How to make sure that no data or properties in the lab are changed by Snow?
    - Lab data are exported from the lab database and copied to a directory on the Snow server. Thus, Snow has no access or possibility to change data in the lab database.
3. How to make sure that the Snow system does not affect the performance of the lab system?

[4] The cell phone number is required in the registration process since the address register does not contain cell phone numbers.

- Verify that the database server can handle more traffic (consider hardware changes if this is not so).
- The number of search agents from the Snow system can be restricted and for instance set to the maximum of one agent.
- The SQL statement can, if supported by the database, use paging and sleep time to reduce CPU consumption.

4. How to make sure that no security threats are transported as messages in the Snow system?
    - Being a message based system makes it hard to impose security threats.
    - Providing content that could possibly be run when starting a new Java process is hard because the startup is parameterized.
    - SQL injection could be a potential threat, but not when proper countermeasures are made.
    - Some parts of the client software may be updated by sending code as part of messages. Such software update messages are signed by the Snow CA and verified by the client before the update is accepted.

8 Server Administration

Openfire server administration is done through a web interface. The XMPP sever administration interface will still be configured, using the firewall and/or web server access control list, to be accessible from local host or a limited set of hosts on a secure port.