Electronic Voting System

Deployment Manual

[1. Introduction 2](#_Toc453123918)

[2. Developing environment deployment 3](#_Toc453123919)

[2.1. Mysql installation 3](#_Toc453123920)

[2.2. Qt IDE installation 3](#_Toc453123921)

[2.3. Subversion server installation 3](#_Toc453123922)

[2.4. Rapidjson library installation 3](#_Toc453123923)

[2.5. Boost 1.54 library installation 4](#_Toc453123924)

[2.6. Openssl installation 4](#_Toc453123925)

[2.7. Deploy self-built Certificate Authority on Local machine 4](#_Toc453123926)

[1.modify the config file of openssl 4](#_Toc453123927)

[2.Build up CA directory 4](#_Toc453123928)

[3.Generate the private key of CA 4](#_Toc453123929)

[4.Generate CA root certificate 4](#_Toc453123930)

[2.8. CA sign the servers’ certificate 5](#_Toc453123931)

[2.9. Database configuration 6](#_Toc453123932)

[2.10. Node.js runtime installation 6](#_Toc453123933)

[2.11. Node.js package installation 7](#_Toc453123934)

[2.12. Configuration of tally servers 7](#_Toc453123935)

[2.13. Edit /etc/hosts for localhost demo 7](#_Toc453123936)

[3. Run under test environment 7](#_Toc453123937)

[3.1. Run Administrator and Voter client 7](#_Toc453123938)

[3.2. Run servers 7](#_Toc453123939)

[4. Delivery of client 8](#_Toc453123940)

1. Introduction

**Server environment:**

We deploy web server based on Linux operating system which can effectively reduce the security risks, in order to ensure the security of the web server. We also use Node.js as Javascript running environment which is a platform based on the V8 engine.

**Database:**

We install the MySQL Community Server (the free version available1) for database server.

**Programming language:**

We programmed the system using C++ and Javascript.

**Library:**

We use Openssl 0.9.8zg, Rapidjson, boost 1.54 and Qt 5.5 as our c++ library.

Moreover, we use the modules listed below as our node.js libraries:

bcrypt-nodejs: 0.0.3,

body-parser: 1.13.1,

connect-flash: 0.1.1,

cookie-parser: 1.3.5,

crypto: 0.0.3,

ejs: 2.3.2,

emailjs: 1.0.5,

express: 4.13.0,

express-session: 1.11.3,

jwt-simple: 0.5.0,

log4js: 0.6.36,

moment: 2.13.0,

morgan: 1.6.0,

mysql: 2.7.0,

node-schedule: 1.1.0

**Version control tool:**

We use svn as our version control tool.

**Data interchange format between C++ and Node.js:**

We use JSON as our data interchange format. JSON is a lightweight text-based data interchange format that is easy for both humans and computers to digest and consume.

**Developing machine environment:**

We use Macintosh as our developing machine and Mac operating system as our developing OS environment.

**Integrated developing environment:**

We use Qt creator and CLion as c++ IDE and Webstorm as node.js IDE. What’s more, we use qmake in Qt creator as the compile script language while using cmake in CLion. Last but not least, we apply npm as the package managing tool for node.js.

2. Developing environment deployment

## 2.1. Mysql installation

http://dev.mysql.com/doc/refman/5.7/en/linux-installation-yum-repo.html

## 2.2. Qt IDE installation

http://doc.qt.io/qt-5/osx.html

## 2.3. Subversion server installation

<http://www.linuxfromscratch.org/blfs/view/7.9/general/svnserver.html>

## 2.4. Rapidjson library installation

<https://github.com/miloyip/rapidjson>

## 2.5. Boost 1.54 library installation

http://www.boost.org/doc/libs/1\_54\_0/doc/html/bbv2/installation.html

## 2.6. Openssl installation

Mac and Linux provide openssl by default, just make sure it has been updated to the newest version.

## 2.7. Deploy self-built Certificate Authority on Local machine

1.modify the config file of openssl

Edit /etc/pki/tls/openssl.cnf file as follow:

[ CA\_default ]

dir = /home/CA #define your CA directory

default\_md = sha256 #at least sha256 is secure nowadays

2.Build up CA directory

mkdir /home/CA

cd /home/CA

mkdir private certs newcerts crl

touch index.txt serial

echo 01 > serial

certs :for cert storage  
crl :for cert revoke list storage  
index.txt : the sequence and belonging of published cert storage  
newcerts : the newly generated cert storage  
serial :inner cert sequence

3.Generate the private key of CA

openssl genrsa -des3 -out private/cakey.pem 2048

-des3 :represent using des3 encryption to protect the CA private key  
2048 :the length of private key

4.Generate CA root certificate

openssl req -new -x509 -key private/cakey.pem -sha256 -out ca.cer -days 36500 -config /etc/pki/tls/openssl.cnf

-sha256 :using hash function sha256 for the generation of root certificate  
-days 36500 :100 years for testing  
-out ca.cer :the keystore to generate ca cert to, which will be imported by client to their browser later on.

## 2.8. CA sign the servers’ certificate

CA signs all the servers’ certificate before system running and we deploy these certificate to the corresponding directories under our tally server, honest verifier server, master server, CA server (for user to apply certificate online).

Here is an example bash script of signing honest verifier server:

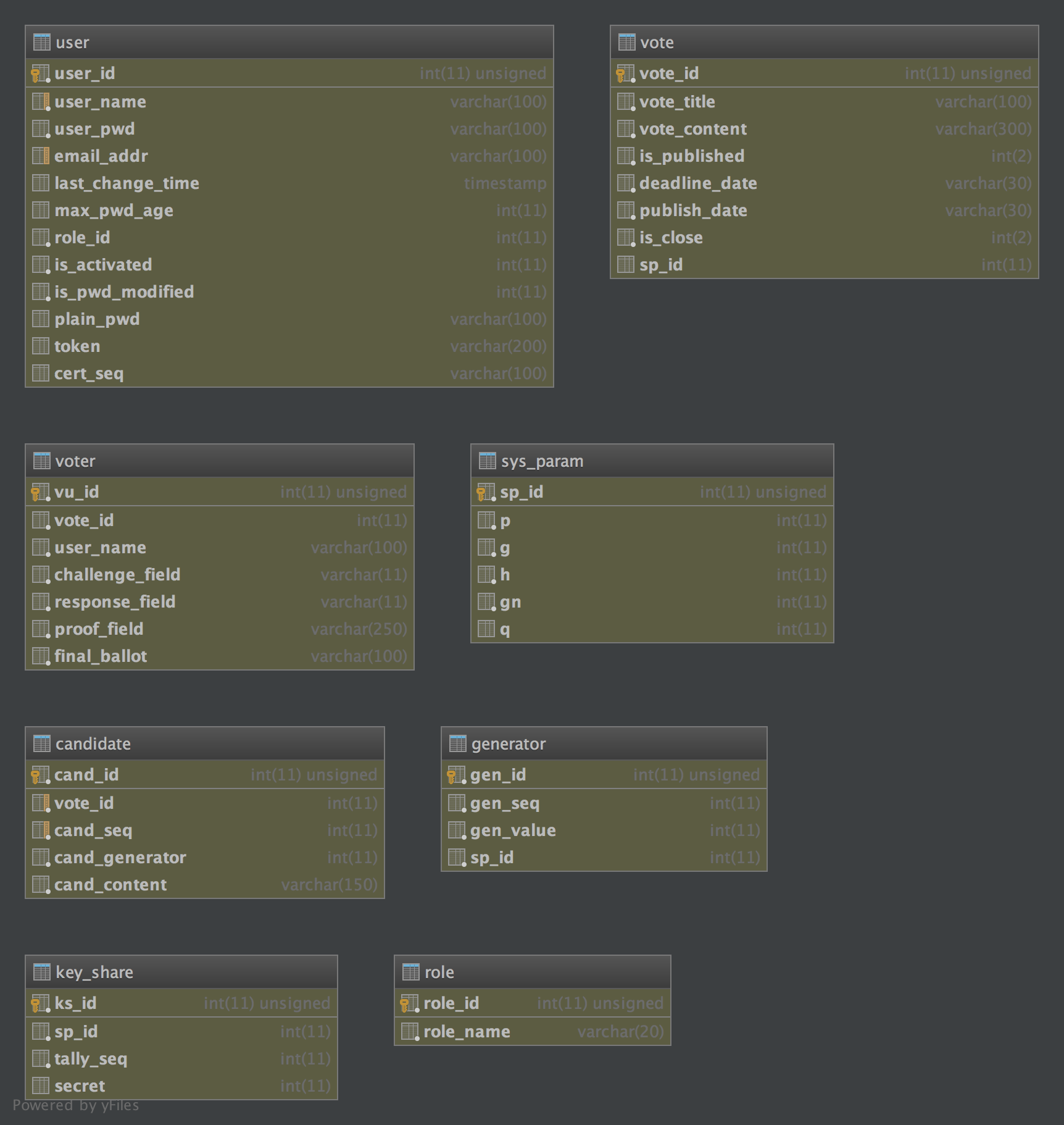
**Create private key and csr used by a server want to apply cert:**

*openssl req -nodes -newkey rsa:2048 -sha256 -keyout honest\_verifier.key -out honest\_verifier.csr -subj "/C=AU/ST=NSW/L=WOLLONGONG/O=UOW/OU=CS/CN=honest\_verifier"*

**Sign a crt for the application used by CA:**

*openssl ca -batch -in honest\_verifier.csr -out honest\_verifier.crt -days 3650 -config /etc/pki/tls/openssl.cnf -passin pass:xxxxxx*

## 2.9. Database configuration



As we mentioned above, we use mysql as our database. We deploy two users for master server and tally server. Tally server do not have access to table “key\_share”, table ”role” and table ”user”, while master server can access all tables.

The database creating script name “table create.sql” is attached under the directory “deployment manual”.

## 2.10. Node.js runtime installation

<https://nodejs.org/en/download/package-manager/>

Easy tutorial to follow step by step, especially on Linux and Mac.

## 2.11. Node.js package installation

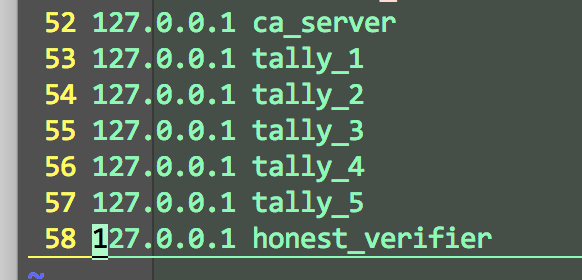
Before running our server, you need to install the modules that we used on local machine.

On linux or mac, use command “cd” to our server like “/path/to/Evoting-Auth-Server/”. Then type in “npm install .”. npm is the “node.js package manager” to read all required modules in “package.json” and install them on local machine.

## 2.12. Configuration of tally servers

There are some parameters of tally servers can be config in “Evoting-Auth-Server/config/tally\_server\_config.js”, and the “tally\_server\_path” field must be config according to actual deployment situation on your machine or remote server.

## 2.13. Edit /etc/hosts for localhost demo



Append these records in /etc/hosts, make sure using “sudo” to edit it. That is because we are using such as “tally\_1” or “ca\_server” as the common name when we applied the corresponding certificate. Without this edition, the server will be treated as “host name is not equal to certificate’s common name” which means the server is untrusted.

3. Run under test environment

## 3.1. Run Administrator and Voter client

Use Qt creator to open project Evoting-client, click “run”.

## 3.2. Run servers

**Deploying Directories of each server on linux or Mac:**

Master server(default port 8080):

“Evoting-Auth-Server/”

CA server (default port 8081, need to be run by root or sudo priviledge):

“Evoting-CA-Server/”

Tally server (no need to startup manually):

“Evoting-Tally-Server/”

Honest verifier(default port 8082):

“Evoting-HV-Server/”

**Usage:**

“cd” to the directories of master server, honest verifier and the ca server, type in the command “node server.js”. You can also define the running port by using “node server.js 8089”.

4. Delivery of client

We can use construction kit to package a released version of client on Mac. Then Qt will deliver a Mac client as we expected. Moreover, the construction method of Windows can be achieve following the tutorial below:

<http://doc.qt.io/qt-5/windows-deployment.html>