Contents

1 Requirements

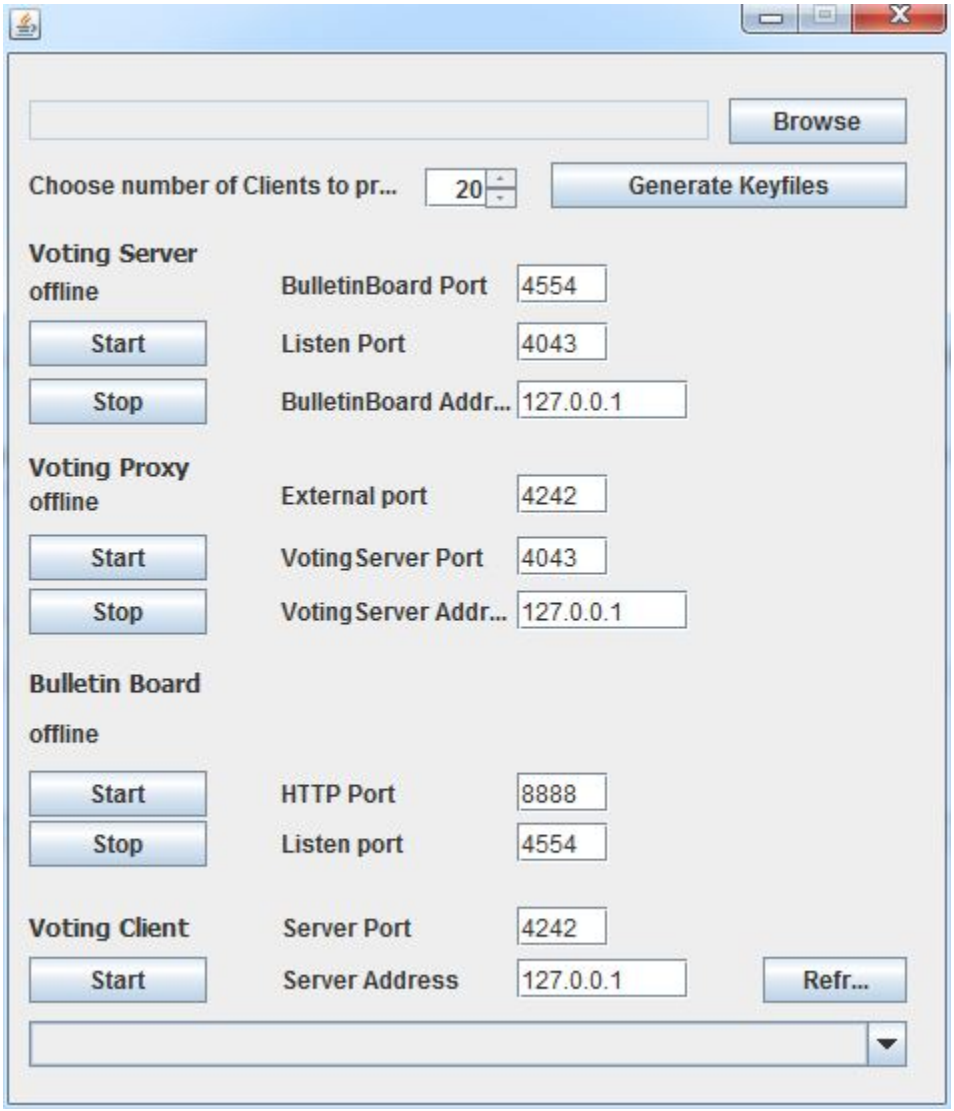
The eVoting Software requires at least Java 1.7 to run, it was tested to work with Windows and Linux (Oracle JRE or JDK). All connection capabilities have been tested with IPv4 connect information.

2 SimplevotingControlPanel

The SimpleVotingControlPanel is a graphical interface for configuring and running all the different components of this software. To start it, navigate to bin directory and use this command (without linebreaks):

```
java -cp .;..\lib\bcprov-jdk16-146.jar
de.uni.trier.infsec.protocols.simplevoting.voterGUI.SimplevotingControlPanel
```

You should see the following window:



2.1 Selecting a directory

First, you must select a directory where the keyfiles are located or to which you want to store generated keys to. Therefor press Browse and select a directory.

2.2 Generating Keys and credentials

In order to run eVoting server or clients you need to generate keyfiles first. If you already generated keyfiles, you can leave out this step, otherwise choose the number of keys to generate and press "Generate Keyfiles". Depending on the number of keys you chose, there will be following files created:

1. Voters private keys (file suffix ".pri"), named voterXX.pri, whereat XX is a subsequent number starting from zero.
2. Voters public keys (file suffix ".pub"), named voterXX.pub, whereat XX is a subsequent number starting from zero. Note: Corresponding public/private keys have same number.
3. Servers key list (contains all voters public keys) named clients.evo

Please note: The servers public and private key are currently included in the source code.

2.3 Controlling VotingServer

To run the eVoting server, select the port to listen for messages from voters or network proxy(**Listen Port**) as well as address and port to send casted ballots and the result to (**BulletinBoard Address / Port**) By pressing Start the server is started. Please make sure the bulletin board is running in order to access the election result.

2.4 Voting Proxy

As VotingServer only serially listens to one port, it might be necessary to enable multiple parallel client connections at one time. For doing this, there is Voting Proxy which can handle multiple connections to its external port and forwards messages to VotingServer. As configuration it takes the external port to listen to (**External Port**) as well as the voting servers connect information (**VotingServer Port / Address**). Voting Proxy does not have any graphical interface.

2.5 Bulletin Board

Bulletin Board is a simple HTTP Server, which accepts the ballots the Voting server casted, as well as the elections results, and makes them public available via HTTP. As configuration it takes the port whereat the webserver listens for GET requests (**HTTP Port**) and the port it listens for messages from eVoting Server (**Listen port**)

2.6 Voting Client

If the selected directory contains public and private keyfiles, a voting client can be started from control panel. Therefor the file must be selected from the dropdown menu. As configuration it takes the connect information of the voting server (**Server Port / Address**). Please note: When using Voting Proxy, the clients are supposed to connect to the proxy!

2.7 General information

When using ControlPanel to start components of the software, each one is started within its own process and Java virtual machine to make sure, there is no interaction of the components beside the network channel. Of course the components are not required to run on the same physical machine.

3 Starting Components without Control panel

You can also start each component without the control panel using these commands:
VotingServer

```
java -cp .;..\lib\bcprov-jdk16-146.jar
de.uni.trier.infsec.protocols.simplevoting.VotingServerStandalone
<path-to-clients.evo> <listen-port> <bulletin-board-address> <bulletin-board-port>
```

VotingClient

```
java -cp .;..\lib\bcprov-jdk16-146.jar
de.uni.trier.infsec.protocols.simplevoting.VoterStandalone
<path-to-publickey> <path-to-privatekey> <voting-proxy-address> <voting-proxy-port>
```

HTTP BulletinBoard

```
java -cp .;..\lib\bcprov-jdk16-146.jar
de.uni.trier.infsec.protocols.simplevoting.HTTPBulletinBoard
<listen-port> <HTTP-server-port>
```

If no parameters are specified, the default ports 4554 for listen port and 8888 for webserver port are used.

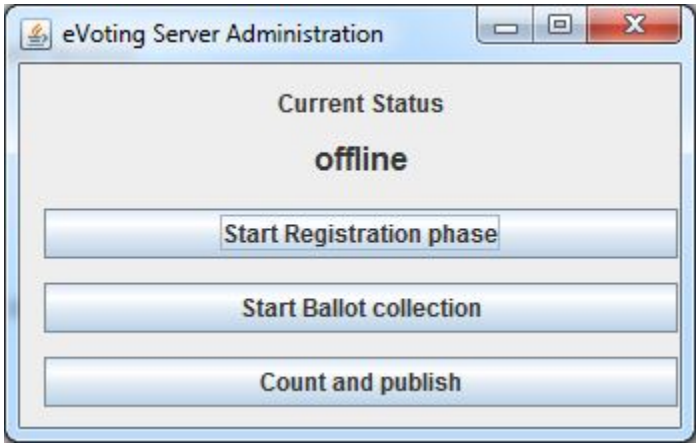
Network Proxy

```
java -cp .;..\lib\bcprov-jdk16-146.jar
de.uni.trier.infsec.protocols.simplevoting.NetworkProxy
<listen-port> <voting-server-address> <voting-server-port>
```

If no parameters are specified, the default ports 4949 for listen port, 4242 for voting server port and 127.0.0.1 for voting server address are used.

4 Voting server

When running Voting Server, this graphical interface will open



Initially the server is offline, which means it does neither accept clients registrations, nor ballot submissions. Note: The server actually does not store its state persistent. Once it is closed, all credentials and ballots get lost.

4.1 Registration phase

By hitting the button **Start Registration phase** the server listens for registration messages. If a valid message is received, it decrypts it, generates credential for the voter and replies a message containing the credential encrypted with voters public key. In case of repetitions, the server will send the stored credential again.

4.2 Ballot Collection

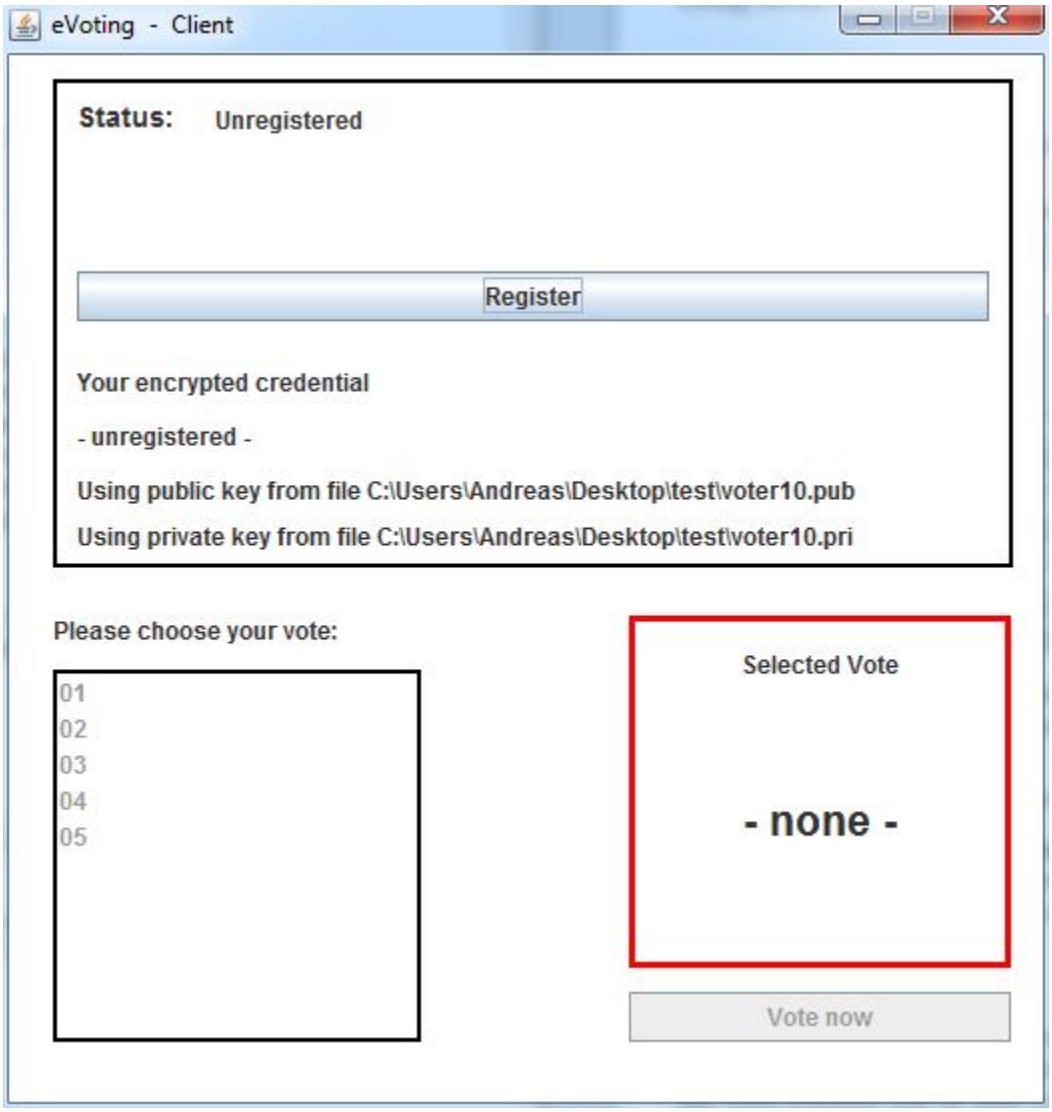
Once the server was switched to collection phase, it cannot switch back. In this phase, it listens for ballot submissions. Only registered clients, which have received a valid credential, can submit their vote. Every ballot that has been received will be sent to the bulletin board and made public available. Note: Revoting policy is "last vote counts".

4.3 Evaluation

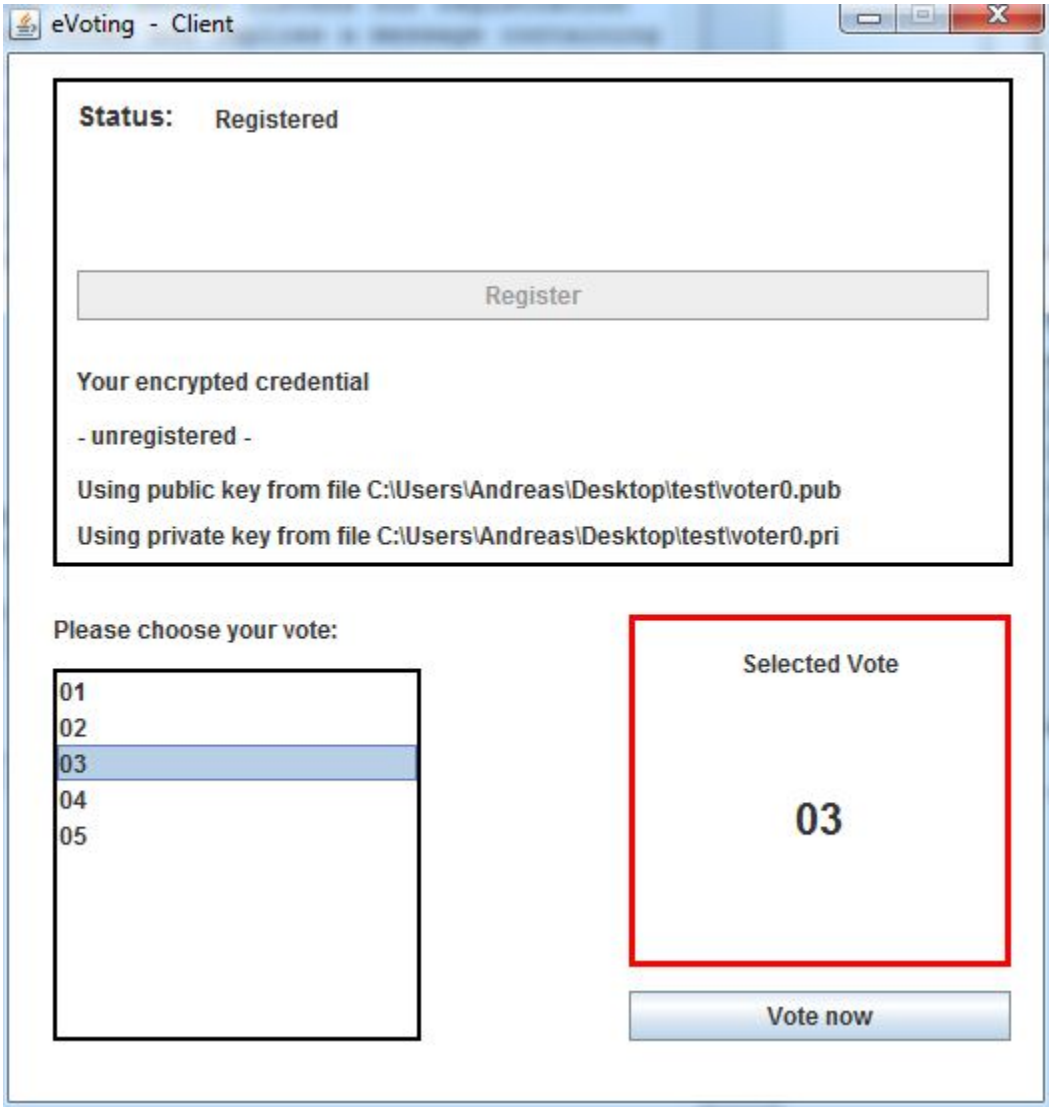
Once **Count and public** is pressed, the server stops collecting votes. All ballots are now decrypted and counted. The result gets published to the bulletin board. It is not possible to switch back to ballot collection.

5 Voting client

When the Voting Client application is started, it looks like this:



As the voting protocol requires registration first, the client will be disabled except the **Register** button. Pressing it, the client will send a request for credentials to the server. If the server is running and is currently in registration phase, it will respond with the encrypted credential. Note: The client stores the credential locally. Once registered, the client can be closed and reopened later. Once registered, the remaining controls get activated:



By hitting **Vote now** the currently selected vote is encrypted with the credential and submitted to the server. Repeated clicks will cause one message per click, only the last vote is casted. You can vote as often as you want, as long as the voting server is in ballot collection phase.

6 Technical information

This software uses BouncyCastle crypto provider to generate keys and perform cryptographic operations. For asymmetric crypto, it uses 1024 bits RSA. Currently, the client and server software expects the clients public and private key as files. These files should have following format:

- Clients public key (.pub files) contain the binary representation of the RSA public key, encoded using ASN.1 encoding, defined in X.509 standard.
- Clients private key (.pri files) contain the binary representation of the RSA private key, encoded using ASN.1 encoding, defined in PKCS#8 Standard
- Servers public Key list format

  First line: Public Key count

  One line per Key, containing the public key in ASN.1 (X.509) encoding represented as hexadecimal string.