

Safe Direct Chat

By X-Princess 51

<http://xprincess51.narod.ru>

User's guide

Disclaimer

SDChat is a free open-source application written on Java. By using the application you agree to the following:

The application is provided for educational purposes as-is with no warranty and usage is on your own responsibility only.

Whatever you do with this application is **on your own responsibility only** and I cannot be held responsible for any misuse or illegal use of this application.

The application is free and you are allowed to modify and distribute it as much as you wish.

If you have questions, you can contact me. However, I don't promise to answer within a reasonable time. The contact information is on my web-page.

My most probable answer to lame questions would be "RTFM".

Advises and improvement offers are welcome. However, I don't promise to implement every one.

If you don't agree with the disclaimer – close this document and erase it from your computer together with the application.

Introduction

The application provides P2P chatting, it means direct connection (peer to peer) between two computers on the network. You can transfer files using this software as well. The application uses AES encryption for data transfer. This encryption system is secure enough even to transfer government information classified as *top secret*. Therefore, SDChat is good enough to protect your private conversation not only from "little sister", but also from a cryptanalizer.

There are many factors which would do everything to spy after you, steal sensitive information about you and eavesdrop on your most intimate chatting. This software can be used for educational purposes and can be used as well by private surfers and organizations who care about their privacy.

This application can work on any operating system with JVM (Java Virtual Machine) installed. The software is easy for use and its communication protocol is very simple. All you have to do is coordinate an encryption key *tete a tete* with your pen-pal and then you can connect to each other and chat or transfer files.

If you find something unclear or unknown terms in this guide, you can ask me or just open Wikipedia.

Hexadecimal base

Every number can be represented in different bases. The base can be anything you wish. The used bases are binary (2), octal (8), decimal (10) and hexadecimal (16). The decimal base is the base you use all the time. Your computer works eventually on binary system. You might not notice it, but the computer eventually has two states: "*there is voltage*" or "*there is no voltage*". The octal base is less commonly used, but the hexadecimal base is used quite widely in computers.

The binary base consists of two digits (0 and 1), octal consists of eight (0-7), decimal consists of ten (0-9) and hexadecimal consists of sixteen (0-9,A-F). In hexadecimal base *A* represents 10, *B* represents 11 and so on until *F* which represents 15. For example number 16 would be 10 in hexadecimal base, 17 would be 11 in hexadecimal and 255 would be FF.

A bit is the smallest data field and can accept only two values (zero or one). Each byte consists of eight bits and can accept 256 values (0-255). A byte can be represented in eight binary digits or two hexadecimal digits. For instance 10111000 or B8 (184 in decimal).

In decimal: $184 = 1 \cdot 10^2 + 8 \cdot 10^1 + 4 \cdot 10^0 = 100 + 80 + 4 = 184$
In binary: $10111000 = 1 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 128 + 32 + 16 + 8 = 184$
In octal: $270 = 2 \cdot 8^2 + 7 \cdot 8^1 + 0 \cdot 8^0 = 128 + 56 = 184$
In hexadecimal: $B8 = 11 \cdot 16^1 + 8 \cdot 16^0 = 176 + 8 = 184$

Another example for hexadecimal number:

$C59F3A = 12 \cdot 16^5 + 5 \cdot 16^4 + 9 \cdot 16^3 + 15 \cdot 16^2 + 3 \cdot 16^1 + 10 \cdot 16^0 = 12582912 + 327680 + 36864 + 3840 + 48 + 10 = 12951354$

Globally it would be like $[digit] \cdot [base]^{[digit\ position - 1]}$ when digit position zero is the ones (when in decimal base), digit position one is the tens, digit position two is the hundreds, digit position three is thousands and so on.

As I have already said, one byte can be represented using two-digit hexadecimal number. Therefore, four bits can be represented using single-digit hex-number. That is why hexadecimal base is used in encryption key settings of SDChat.

AES encryption

AES stands for *Advanced Encryption Standard*. It is performed using *Rijndael* encryption algorithm, which is considered by United States government as secure enough. There is another encryption algorithm called *Serpent*. Serpent is considered as a little more secure than Rijndael, but Serpent is much slower as it performs 32 rounds (while Rijndael performs 10-14 rounds). Still, Rijndael is secure enough for top secret data and that was why US government prefers Rijndael as AES algorithm.

How AES works

Encryption algorithms usually use an *encryption key*. The DES (Data Encryption Standard) is an old weak algorithm which uses only 56-bit key (seven bytes). AES uses 128-bit, 192-bit or 256-bit keys (16/24/32 bytes). Entities like US government usually work with 256-bits key.

The algorithm works with blocks of 128 bits, meaning it encrypts your data by sixteen bytes.

First, the algorithm creates a *round key*. A round key is an array of 128-bit "keys" produced from your encryption key and every array member is based on the previous one in the array. The first array member (or two members, depending on key length) is of course your key. A different member is used for every encryption round. The algorithm performs ten rounds for 128-bits key, twelve rounds for 192-bits key and fourteen rounds for 256-bits key. An *encryption round* is actually a single combination of encryption steps.

AES steps include non-linear substitute of the block using a table of constants for each byte, linear mixing of the block (columns mix) and XOR operation of the block with a member of the round key.

XOR: *Exclusive OR*. An operation between two bits that gives the following results:

Bit 1	Bit 2	Result
0	0	0
0	1	1
1	0	1
1	1	0

Each round performs all three steps, except for the last round which skips the columns mixing operation and the first round which only adds the round key.

Encryption systems requiring conceal of the algorithm worth nothing. Systems like that can only protect your privacy from "little sister". Encryption application must assume the enemy is a strong computers freak who can use disassembler and learn the algorithm. AES (Serpent and TwoFish as well) answer this requirement. These algorithms are known to every fool and this makes the encryption systems based on them secure even against cock-suckers (sorry for the expression) from the "thoughts police" who eavesdrop on your most intimate conversations. So SDChat is an open-source application source code of which can be seen by any asshole.

The encryption key of SDChat is set using hexadecimal system. Each hexadecimal digit sets four bits. Therefore, 128-bit key would consist of thirty two digits, 192-bit key would consist of forty eight digits and 256-bit key would consist of sixty four digits. To be secure, the encryption key must be as random as possible (SDChat provides a key generator for that purpose).

The application assumes that the god and devil damned thoughts police has full control of all communication media, including yours. SDChat protects your chat from eavesdropping, as long as you keep your computer and key secured. The perfect way to secure a key or password is to memorize it. I know it is very difficult to memorize a sequence of sixty four hexadecimal digits, but a user who really wants to secure his or her data would memorize even that. For the lazy users, SDChat provides an ASCII password converter which converts an alpha-numeric password to encryption key. It is much less secure as alpha-numeric characters cannot cover all the possible 256 values of a byte (however you can modify a converted password before saving it as the default key). For the real lazyheads, SDChat provides a possibility to save the default encryption key in a configuration file, but use this feature only if your computer is secured enough against unauthorized access.

Every encryption, at least theoretically, can be broken. If not by mathematical cryptanalysis, then by brute force (not real, but theoretically possible). Therefore, you are strongly advised to change the encryption key from time to time. The best is to do it every time you meet your pen-pal tete a tete.

Side-channel attack

There is one thing SDChat (and any other secure line) cannot protect you from at all, and it is *side-channel attack*. The worse thing is that if mother fuckers from thoughts police want to eavesdrop on you, they will probably use side-channel attack rather than trying to fight SDChat using mathematical cryptanalysis or brute-force which may take billions of years (and I am not exaggerating).

The simplest example of a side-channel attack is just hacking your computer and gain access to your RAM or even full control of your computer and just see the contents of your chat. This is especially relevant to Micro\$oft Windows users, because even god and devil do not know which contracts Micro\$oft made with entities like FBI (especially after 11/09/2001). This is less relevant for users of open-source systems because finding and fixing security holes in open-source systems is a lot easier and can be done by every simple user.

Another example of side-channel attack is some son of a bitch gaining access to your computer for a few seconds when you are not looking, installing a key-logger and seeing all the messages you type.

Side-channel attack does not necessarily have to be hacking. You can for example sit on a park bench with your laptop and chatting with your pen-pal via a public WI-FI. During that, some sucker might hide behind your back and see all your chat (or hide somewhere on a skyscraper's roof and watch you and your laptop's screen through binoculars).

One more extreme example of side-channel attack is capturing radiation from CRT monitor using very sensitive device, rebuilding the picture from the radiation and seeing your chat.

You have been warned! Nothing and no one would be able to protect you, unless you keep your computer extremely secure.

There are many applications on the market that offer encryption. To protect your files, I would recommend *TrueCrypt* (<http://www.truecrypt.org>).

If you want a powerful, secure, stable, free, friendly and reliable operating system – I would recommend *Ubuntu* (<http://www.ubuntu.com>).

If you have any questions, you are free to contact me and I'll be glad to help.

Beware that a shit-eater from thoughts police can also gain the encryption key by force such as bulgury, bribery, threats/extortion (for instance psychological coercion), tortute (also called *rubber-hose cryptanalysis*) and social engineering. You have been warned!

Easter egg

There is one more thing that you an your pen-pal need, which must be kept even more securely than the encryption key, and it is *easter egg*. It should be used in the beginning of the secure conversation in order to make you sure that you really talk with your pen-pal and not with some dickhead from thoughts police who stole or robbed your pen-pal's computer. It works simply: you ask your pen-pal a normal question, but he or she must give a special answer.

For instance:

You: "Hi, what's up?"

Your pen-pal: "Cool, what about you?"

You: "Nice. Tell me, what do you like to eat?"

Your pen-pal: "No matter what, as long as the food comes with plenty of ketchup."

In this example, if your pen-pal answers something like "hot dogs" or "hamburgers", then you know it is not your pen-pal but some shithead from thoughts police. You can also make special "trouble signal" – for instance if your pen-pal answers "pizza" then you know it is your pen-pal and some sucker from thoughts police pointing a gun at him or her and forcing him/her to make a "little red riding hood" to you (it means make you tell all your secrets while you think you chat with your pen-pal, but the thoughts police sees all your chat).

Coordinate your easter egg only tete a tete and do not write it anywhere! Memorize it! Moreover, use the easter egg only in secure chat, nowhere else. If you care about your privacy, assume all communication lines (phones, radio, satellite-communication, email, Facebook, ICQ and so on) are fully controlled and monitored by the thoughts police.

TCP/IP

After talking about information security, let me give you a small introduction to the technical part of Internet communication.

TCP/IP is the Internet communication protocol. TCP stands for *Transmission Control Protocol* and IP stands for *Internet Protocol*.

Like a telephone has a number in phones network, every computer has a logical address in a network of computers. It is called "IP address". The forth version of IP address consists of four octates. Each octate is a number from zero to 255, so each IP address consists of thirty two bits. Usually IP address is written as "X.X.X.X" where X is a number from zero to 255.

Example for IP address: *194.67.1.14* (you can try entering it in your browser).

Known IP addresses:

127.0.0.1: A loopback address. An internal address of the computer, so when the computer sends data to this address, it actually sends it to itself.

169.254.X.X: Internal address of Micro\$oft networks. The computer also sets address like that to itself when it fails to acquire an IP address from the network.

10.X.X.X: Internal network class A (can hold up to 16777214 computers).

192.168.X.X: Internal network class C (can hold up to 253 computers).

Another part of the IP data is the *subnet mask*. It determines which bits belong to the network address and which bits belong to the stations' addresses. The subnet mask of class A is 255.0.0.0 and the subnet mask of class C is 255.255.255.0 .

In case of class A, the first octate is a part of the network address (which shall be 10.0.0.0) and the other three octates are computer addresses. In case of class C, the first three octates belong to the network and the last octate belongs to computers. The last address of the network (X.X.X.255) is reserved for broadcast.

The *default gateway* means the connection to the WAN (*Wide Area Network*). For instance your LAN (*Local Area Network*) is your home network or the school network and the WAN is the Internet. The default gateway is usually your router or can be a special server which connects you to the outside world. The following table shows examples of routers and IP data:

Router	Computer's IP address	Subnet mask	Default gateway
Cable Ambit router	192.168.0.100	255.255.255.0	192.168.0.1
ADSL router	10.0.0.1	255.0.0.0	10.0.0.138
Edimax	192.168.2.100	255.255.255.0	192.168.2.1
TPLink/Netgear	192.168.1.1	255.255.255.0	192.168.1.1
LevelOne	192.168.123.100	255.255.255.0	192.168.123.254

DNS system

Every website you surf is stored on some server. Have you ever wondered how you enter something like "www.walla.co.il" and reach the web-site? This is done using the DNS system. DNS stands for *Domain Naming System*. It works using DNS servers which addresses are defined in your computer's IP configuration. Try entering "<http://192.118.82.140>" in your browser.

Everytime you enter a hostname (name of a site), your computer sends request to the DNS server and gets needed server's IP address. After that your computer connects to the IP address and you can surf the website.

DHCP

Most of the networks use automatic IP configuration for computers. It is done using the DHCP system. DHCP stands for *Dynamic Host Configuration Protocol*. Every time your computer connects to a network, it sends a broadcast with DHCP request (unless it is set to use static IP) and gets a reply from the DHCP server with the IP configuration. In most cases, the network DHCP server is the router.

TCP

The TCP protocol is a very important part of the IP protocol. It provides a reliable communication between computers. "Reliable" means that TCP guarantees your data to arrive at the remote computer correctly and in the same order it was sent. This guarantee is mainly achieved by packets enumeration and CRC hashing (errors check).

The TCP connection is established using three-way handshake. It means your computer sends a request to the remote side, the remote computer replies with authorization and then your computer sends the third synchronization data packet.

There is a second protocol called UDP. It stands for *User Datagram Protocol*. This protocol is "launch and forget". Your computer just sends a packet with data and does not care whether the packet reached its destination or not and whether it reached the destination correctly. This protocol is good for simple checks or for implementing voice/video communication over the Internet.

TCP and UDP protocols use virtual *ports*. The port range is from 1 to 65535. When an application wants to connect to another application at a remote server (for instance your browser to the web-server), it must specify a port among the connection data.

In order to accept a connection, an application must listen on a specified port. Otherwise, the system won't know which application shall receive the data. This is why ports are used. Let's take a web-server for example: it listens on port 80. Your computer connects to port 80 on the server, sends request and receives a web-page as a reply.

Known default ports:

- 80: TCP port used for HTTP (*HyperText Transfer Protocol*) connections, like web-sites .
- 21: FTP (*File Transfer Protocol*).
- 23: Telnet. Used to connect remotely to another computer's console.
- 22: SSH (*Secure Shell*). Like telnet, but secured.
- 25: SMTP (*Simple Mail Transfer Protocol*). Used to send email.
- 110: POP3 (*Post Office Protocol*). Used to receive email.
- 443: SSL (*Secure Socket Layer*). Used mainly in secure web-sites, like banks.
- 445: Used for Windows shares connections.
- 53: UDP port used to resolve hostnames (DNS).

There are many other known ports which I am not going to list here.

Just like your computer connects to web-server, the same happens in SDChat. Your pen-pal specifies a port to listen (38295 by default) and activates the listener. You open a chat window, enter his or her IP address (you can enter hostname as well), enter the port number and connect. Your pen-pal's computer accepts the connection and knows to pass the data to SDChat client and you can communicate.

Just like incoming connection is performed to specified port, outgoing connection is also done from a specified port. The port for outgoing connection is usually random and can be used only for current connection until closed.

Firewall

A firewall is actually a traffic filter and used for securing computers on a network. It can be hardware (Checkpoint router for example) or software (ZoneAlarm or Outpost for instance). A basic firewall monitors all the incoming and outgoing network traffic in the computer, meaning all the traffic passes through the firewall.

Once the firewall has been installed on a system, the user must set rules according to which the firewall decides whether to allow specified traffic or drop it. This is how firewall stops hackers, trojan horses, spywares and script-kiddies. Let's take the ancient NetBus trojan for instance: NetBus listens on port 12345. In order to take remote control of your computer, the attacker must connect to port 12345. If you have a properly configured firewall, then the packets received from the attacker are dropped and he or she cannot connect to you.

Every computer must have a firewall. If you are using Micro\$oft Windows, then there are many firewall systems like ZoneAlarm or Outpost. Besides, modern Windows systems have internal firewalls. If you are using Linux, then it contains firewall and you should install *FireStarter* which is a friendly application to configure the firewall.

You may ask how protected computers are hacked, and I'll tell that nothing is perfect. The most important feature about firewall is robustness. Perfect programs with no bugs exists only in theory. Every system can be hacked, it is just a matter of time. The time to hack a system may be a few seconds and may be a trillions of years. The same is about cryptography.

Besides you need an anti-virus, especially if you are a capitalist running DOS (or Micro\$oft Windows user in official words). Linux and Macintosh are very strong and secure systems, therefore writing viruses for these systems is a suicide. Still, there are a couple of viruses that can do very little harm if you are stupid user. If you use Linux, you can install Clamav, an open-source anti-virus for Linux. There are other free or partially free anti-virus software for other systems like AVG and Avast.

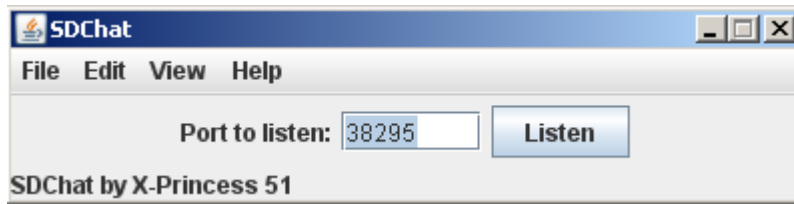
My personal advise: use only free software (I mean community created open-source software, like Linux) and pay nothing to capitalists, especially to anti-virus producers. There are plenty of rumors they produce viruses in order to improve their sales. There are also many rumors that governments produce and distribute different trojans in order to spy after civilians, but anti-virus producers don't want to make exclusions for governments.

SDChat application

How to run the application

In order to run the application you need Java installed on your computer. You can download the Java machine from <http://www.sun.com> and install it very easily. Once you've installed Java, all you have to do is open SDChat folder and execute "run.bat" (works both in Windows and Linux).

In case the script does not work, you can just run the file "SDChat.class" using the Java machine.



Once you've executed SDChat, you shall have disclaimer window. If you don't agree with the disclaimer, click "No" and the application shall quit. If you do agree, click "Yes" and you shall get the application.

The main window

In the main window you have menus, listening panel and the status line. The status line (where you see "SDChat by X-Princess 51" on the screenshot) shows you different application messages.

Port to listen: This is where you shall enter the local port SDChat should listen on for incoming connections.

Listen: Click this button to start (or stop) listening for incoming connections. Don't activate listener unless your pen-pal is supposed to connect to you right now. Also, due to security reasons, once a connection is accepted, SDChat stops listening and if you want to accept another connection you have to activate the listener again.

When the listener is activated, your pen-pal shall connect to the port with a chat or file send request.

Menus

In the "File" menu you shall find functions to open chat or file transfer (for connecting to remote side). In addition you can find options to load or save configuration file and of course to exit the application.

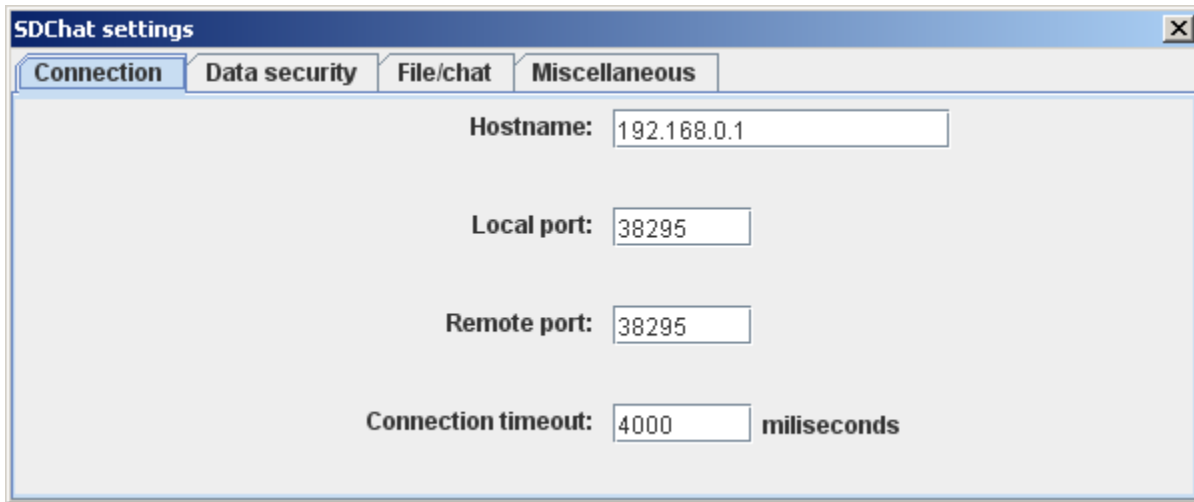
Under the "Edit" menu you can find key generator and the password converter.

In the "View" menu you can open the settings window.

In the "Help" menu you can see the "About" screen.

Configuring SDChat

SDChat provides different configuration tools. The main configuration tool is the settings window. To access the settings window go to "View" menu and open "Settings" (or press F4 from the main window).



In the settings window you have four tabs: "Connection", "Data security", "File/chat" and "Miscellaneous". The settings are saved automatically when you close the window using the window close button or <ESCAPE> key. If you do not want to save settings, click on "Discard settings" on the "Miscellaneous" tab (or press CTRL+Z). Most of the functions have hotkeys. Just hold your mouse over the function to see the tooltip with the hotkey.

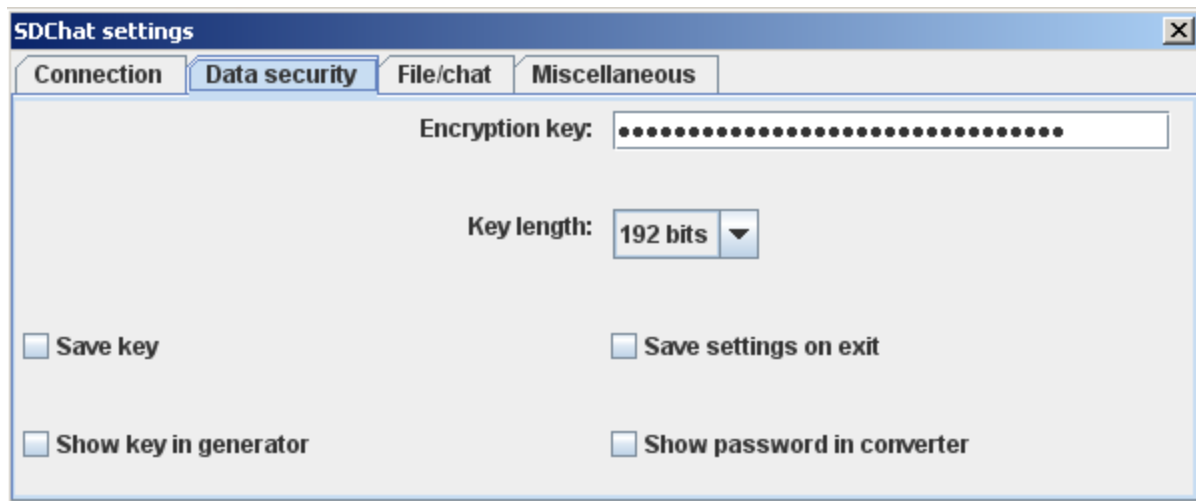
In the "Connection" tab you can configure default remote hostname, default local/remote port and outgoing connection timeout.

Hostname: You can enter default IP address or hostname of your pen-pal which will appear in the hostname field of outgoing chat and file send requests. Especially useful when your pen-pal has fixed (permanent) IP address.

Local port: Accepts a value from 1 to 65535 and is used as the default port to listen to which appears in the main window.

Remote port: The listening port of your pen-pal which shall appear by default in outgoing chat and file send requests.

Connection timeout: The time in milliseconds to wait for replying synchronization packet from your pen-pal after the first packet of the three-way handshake was sent. Can be from 4000 to 15000 milliseconds (it means four to fifteen seconds). If the connection times out, then it is not established.



In the "Data security" tab you can define the default encryption key, the default key length for key generator, key saving flag in configuration files, default key displaying flag in key generator, saving settings on exit flag and default password showing flag in password converter.

Encryption key: The default encryption key. Can be 32 digits (128 bits), 48 digits (192 bits) or 64 digits (256 bits). If you enter less than 32 digits, the key is completed to 32 digits with zeroes. The same happens if you enter 33-47 or 49-63 digits (the keys are completed with zeroes to 48 or 64 digits). The key is hidden when not focused and revealed when focused. The default key appears by default in key fields for outgoing chat and file send request, but also used as the encryption key in incoming connections. It means that if your pen-pal connects to you, then you must set the encryption key in the settings window. Every encryption key field of SDChat works like the one in settings window.

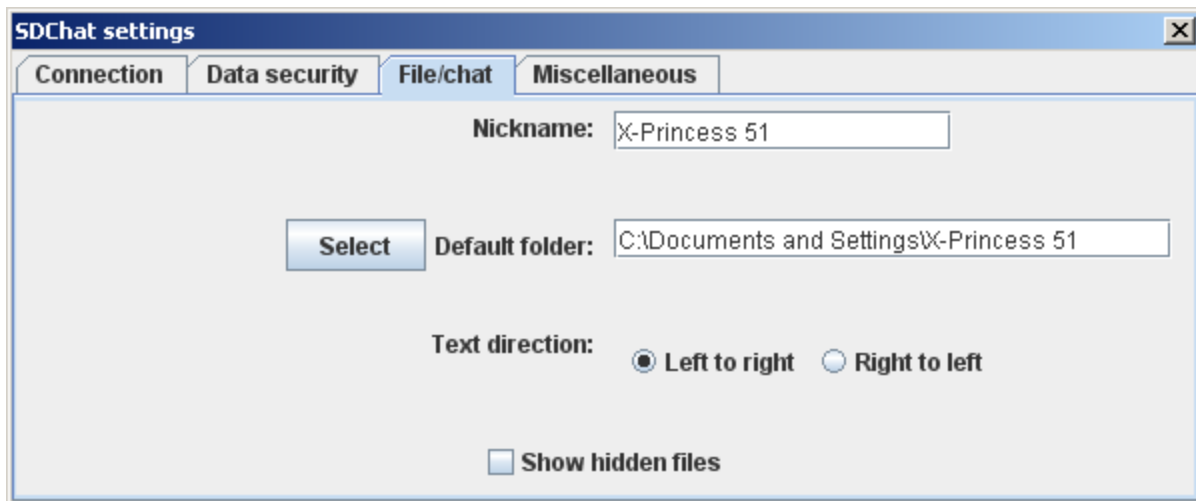
Key length: The default key length to use in random key generator. Can be 128, 192 or 256 bits. You can of course change the key length in the generator window itself, but it won't be saved into the configuration.

Save key: A flag which if checked – saves key into configuration file (on exit if allowed, or to custom configuration file). You should check this option only if your computer is secure enough and no one else has access to it.

Show key in generator: This flag determines whether the generated key is visible on screen or not. If yes, the generated key is shown in the middle of the key generator window in red digits. Like the key length, this flag can be changed in the key generator window, but then it won't be saved into the configuration.

Save settings on exit: The application offers a default configuration file "SDChat.cfg". All the settings are saved into it and loaded next time you run SDChat, unless you uncheck this option. If this option is unchecked, the only data saved into the default configuration file is not to save settings on exit.

Show password in converter: This flag determines whether to reveal the password while typing in the converter or to keep it hidden.



In the "File/chat" tab you can set your nickname, the default folder to load or save files, the default text direction in chat windows and whether to show hidden files by default while selecting file or directory.

Nickname: This is your nickname used in chat and outgoing file transfers. The nickname can accept any characters, but no more than than forty eight.

Default folder: This is the folder the application first opens when loading of saving files. By default this is your home folder. The folder you select must exist an accessible.

Text direction: While chatting, you can write and see the chat from left to right (English for example) or from right to left (Hebrew for example). You do it using CTRL+SHIFT in the line where you type your message or in the chat text area. Here, in settings window, you can set the default text direction (LTR or RTL).

Show hidden files: This flag sets whether to display or hide hidden files by default in file/folder selecting windows.



In the "Miscellaneous" tab you can discard and changes you made, restore application's default settings and choose whether to show disclaimer at application start-up.

Discard settings: When you close the settings window, all the changes are saved. If you click this button, the settings are returned to the state before you opened the settings window.

Restore defaults: Reset all settings to their default values.

Show disclaimer: Every time you run SDChat, a disclaimer window is shown at the beginning. If you get fed up with that, simple uncheck this option (and of course "Save settings on exit" must be enabled) and no disclaimer would be shown at SDChat start-up.

Load/save settings

You can save your configuration file and then reload it later, independent of the "Save settings" flag. To load configuration file use "Load configuration file" from "File" menu of the main window and to save your configuration file use "Save configuration file" from the same menu.

The configuration load function is disabled while the client is listening for incoming connections.

Key generator



The key generator provides you an option to generate a completely random encryption key. All you have to do is open the key generator, move your mouse within the window as long and as randomly as possible and then to save the generated key as the encryption key.

How the key generator works

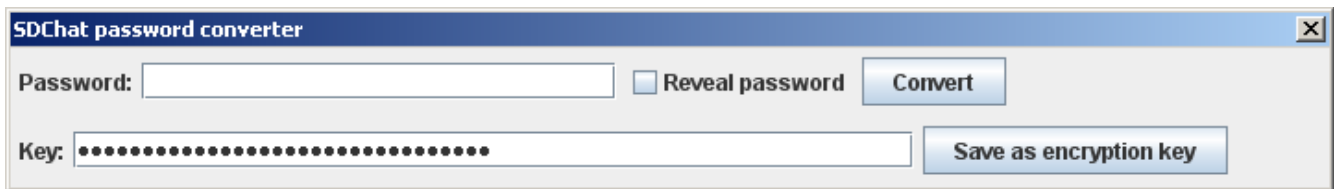
Each time you move your mouse within the generator window, mouse coordinates are read. Besides a pointer runs digit by digit over the key again and again every time you move your mouse. If the pointer is even, X coordinate is used. If odd then Y coordinate is used. Then the system calculates the remainder of the coordinate divided by sixteen and this is how the hexadecimal digit is generated. The more you play with your mouse within the generator window, the better key you get.

Save as encryption key: Once you have generated the key, click this button to save the generated key as the default encryption key. Then you can exchange the key with your pen-pal tete a tete and enjoy.

Key length: You can choose the length of the randomly generated key. Of course it can be 128, 192 or 256 bits.

Show key: Check this option if you want to see the generated key as you generate it. The key appears in red in the middle of the window.

Password converter

The image shows a window titled "SDChat password converter". Inside the window, there is a "Password:" label followed by a text input field. To the right of the input field is a checkbox labeled "Reveal password". Further right is a button labeled "Convert". Below the password field is a "Key:" label followed by a text input field filled with 24 dots. To the right of this field is a button labeled "Save as encryption key". The window has a standard Windows-style title bar with a close button (X) in the top right corner.

The password converter can help a user who prefers not to save the encryption key anywhere, but also does not want to memorize a key. It is much easier to memorize an ASCII password. Simply type your password into the password field, click "Convert" and then save it as the default encryption key.

Password: This is where you shall enter your password. The password can be only ASCII and each character means a byte, meaning the password should be 16/24/32 characters. If less, the rest of the key is completed with zeroes (just exactly if you enter an incomplete key).

Reveal password: Sets whether to show the password while typing or still hide it. The password is hidden anyway when you leave the field.

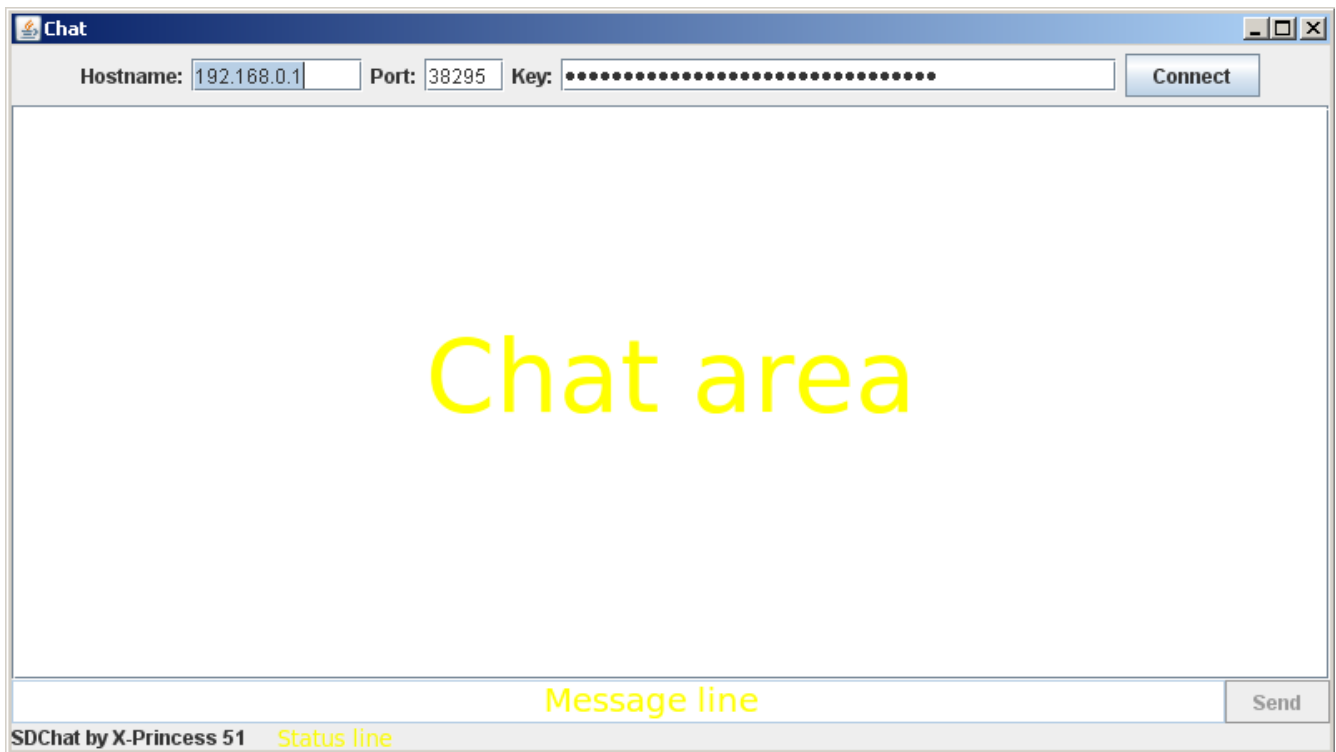
Convert: Click this button to convert the password you entered into an encryption key. The key is set in the "Key" field below.

Key: The resulting from password encryption key. Before conversion, your default encryption key is set there.

Save as encryption key: Click this button to save the encryption key in the "Key" field as the default encryption key.

If you close the password converter without saving the key, nothing is changed in SDChat configuration.

The chat window



The chat window is the part of the application allowing you to text-chat with your pen-pal. All you have to do is enter his or her IP address (or hostname), remote port, encryption key and then click "Connect".

Hostname: This is where you should enter the network address of your pen-pal.

Port: The port your pen-pal's SDChat client is listening on (or in other words "Remote port").

Key: The encryption key to use during the chat. The chat window suggests your default encryption key, and allows you to change it for the current chat.

Connect: Click here to connect and start chatting with your pen-pal (or disconnect).

Send: Click here to send your message to your pen-pal. This buttons is enabled only when you are connected. Empty messages are not allowed. You can also press <ENTER> from the bottom line where you type your message.

Chat area: This is the large part of the chat window which displays your chat contents. It displays both your messages and your pen-pal's messages. You can change text direction by pressing CTRL+SHIFT when the chat area is focused.

Chat history – there is no such thing in SDChat, not due to laziness but for data security reasons.

Message line: This is where you type the message you wanna send to your pen-pal. It is only enabled while you are connected. The chat works with unicode, so you can type any displayable characters in your message. Changing text direction is CTRL+SHIFT. To send the message immediately after typing, you can simply press <ENTER> from the message line.

Status line: Shows chat notifications like connection loss. Also notifies you when your pen-pal is typing a message.

Once you have entered IP, port and key, just click on "Connect" or press <ENTER> from one of those fields. Your system will try to connect. You are notified if any error occurs. You won't be able to control the chat window until it is connected or an error/timeout occurs. The connection timeout is set in application's settings. 4000 miliseconds should be good for a normal stable network. Once you are connected, it is very recommended to use the easter egg and then to start chatting securely. **Remember: easter egg must be even more secret than the encryption key!**



If your pen-pal connects to you, then as I said, the encryption key must be set in SDChat settings and you shall click "Listen" button in the main window (with the correct port of course). Once you've done it, SDChat starts

listening on the given port. When someone connects to you, SDChat immediately closes the listening port due to information security reasons and waits for "greeting block" from your pen-pal. That block tells whether it is a chat or file transfer. If a chat – SDChat simply opens an already connected chat window and you can have a secure conversation.

The encryption key must be correct to the last bit. Not due to laziness but due to information security reasons SDChat makes no key verification and no hashing. If even single bit of the key differs from your pen-pal's key – in the best case you would see scrawl from pen-pal's side instead of messages (in worse cases SDChat may crash or hang-up).

Also, make sure you connect to the correct IP address and receive connections from the correct IP (you see your pen-pal's IP while connected). You can coordinate IP address even on unsecured lines because the thoughts police probably knows it anyway. Therefore, that information's value is usually no higher than onion husks (assuming your computer is secured enough). Still, for extra security, I highly recommend to use what is called "human codes". For instance ask your pen-pal "What are your bets?" and if his or her IP is 192.118.82.140 (just for example), he or she shall say "My bets are 192, 118, 82 and 140".

Chat – behind the curtains

In this chapter I'm going to explain how the chat works, including the simple communication protocol.

The communication protocol works in blocks of sixteen bytes rather than single bytes, that's in order to allow data be encrypted using AES. All data is sent block by block and each block is encrypted before being sent and decrypted after being received.

I am giving you the information assuming you and your pen-pal's both encryption keys are correct and your communication line is free of distortion (meaning that the data arrives with no errors). **Always assume your line is constantly monitored by the thoughts police.**

"Greeting block"

When you connect to your pen-pal, the first action performed is sending the "greeting block". Like every block, it consists of sixteen bytes. As you already know, every byte can accept 256 values. Every two bytes create a *word* (can accept 65,536 values). Every four bytes create a *double word* which can accept 4,294,967,296 values. Every eight bytes create what I'm gonna call here *long* which can accept 18,446,744,073,709,551,616 (or 2^{64}) values. In our case it would be an integer number up to 9,223,372,036,854,775,807 (or $2^{63}-1$).

The greeting block is actually two longs. The first long is a random integer number with only one rule: if it is even then it is a chat and if it is odd then it is a file transfer.

In case of chat – the second long is a completely random number. In case of file transfer – the second long is the size of the file in bytes.

Nickname

Immediately after your client sends the greeting block, it sends your nickname in a way of a text. Once your pen-pal's client receives the greeting block, it also sends you his or her nickname to you.

The nickname is sent in the same way a message is send, but interpreted as a nickname due to being the first text received.

Messages

Each message is sent as unicode, including nicknames. In that way, every character takes two bytes, meaning a block contains eight characters. When you send your message, a special non-displayable character unicode value of which is zero, added at the end of the message. Every character has a numeric value. That characters means end of message. Once the zero character has been added, the message is started being send in blocks, meaning the message is send by each eight characters. If the last block of the message is larger than needed, it is filled with random bytes after the zero character.

The remote side listens for incoming data during the connection and stores all the characters from received blocks. When it encounters the zero character, it simply displays the message and of course resets the buffer. The length of the message is limited to 4096 characters.

If sequence of more than 1024 message blocks with no zero character are received, it is interpreted as flood-attack attempt and the connection is immediately broken.

Typing notification

When you pen-pal is typing a message, you shall see a notification in the status line.

Each time you type within the message line, a special block is send to your pen-pal. The block consist of completely random bytes except for one: one of the eight characters is <ENTER>. Its place within the block is also random.

The application of course does not send the block for every key-click. When a key is pressed within the message line, SDChat, besides sending a block, activates a countdown of a couple of seconds. Next time you press a key and the countdown is still running, no block is sent. Only once the countdown reaches zero, the notification block can be sent again.

When the remote side receives a type notification, it displays the message in the status line and activates a countdown of a couple of seconds. Once the countdown reaches zero, the message disappears.

Due to the rule of <ENTER> within notification block, message blocks never contain the <ENTER> sign. If random data has to be generated within message block, the application makes sure no one of the characters is <ENTER>. If is – the character is simply increased by one.

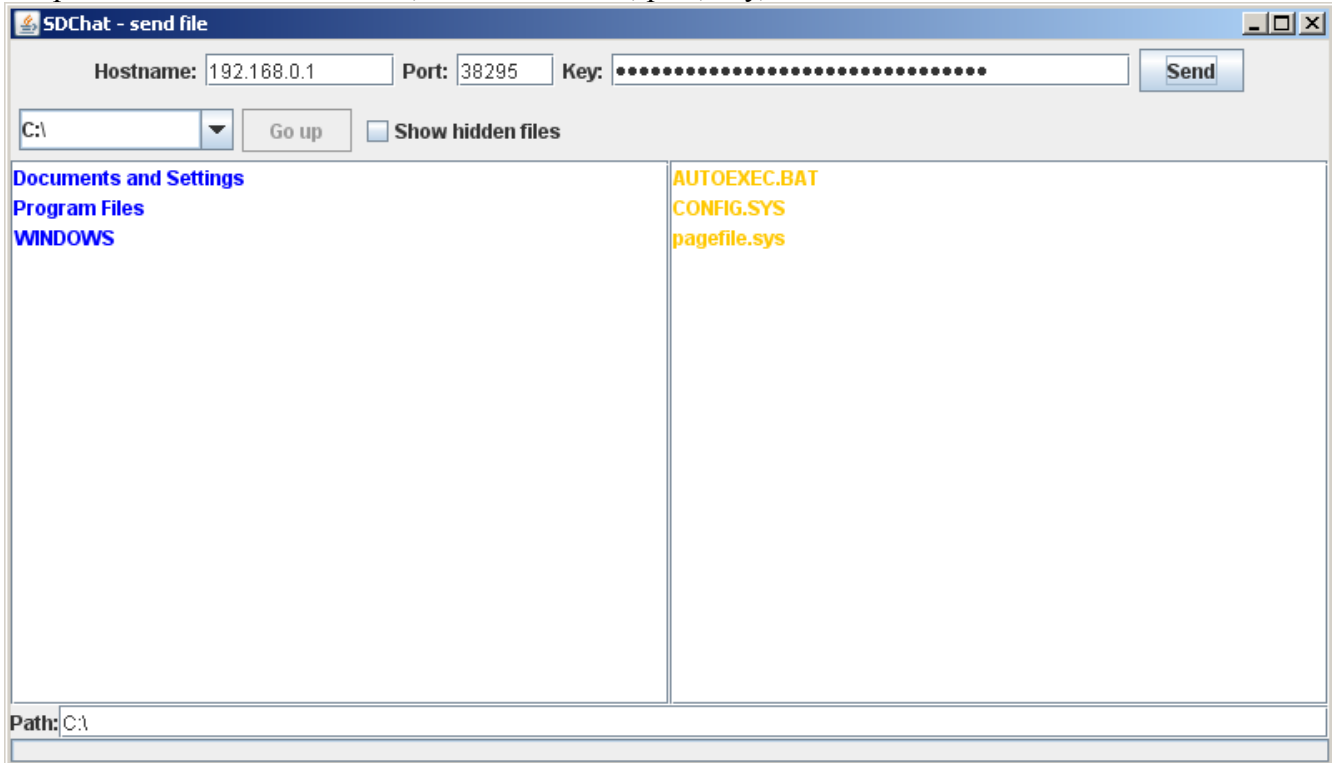
Important

As I've already said, each character is two bytes. The less significant byte changes with every character, but the more significant byte usually remains almost the same. If your chat is in English, the more significant byte is always zero. It means that every odd byte in the block is zero. If you chat in other languages, the more significant byte does not change much as well. This is the way the thoughts police can make good assumptions which bits of your blocks are the same and theoretically it might make its crime against humanity in breaking your cipher a little easier. The block is of course heavily encrypted, but in this way the thoughts police may know a part of the result which might help it a little bit in doing its crime.

There is no magical solution for every problem, but the simplest solution for many problems is to **change the encryption key as often as possible.**

File transfer

Besides chatting, SDChat provides an option to send files via the secure channel. All you have to do is to open the file transfer window, enter IP address, port, key, choose file and send.



This is the first window you see once you open file transfer. Just like in chat window, you have to specify hostname, port and key (offered automatically from application settings).

Once you have entered that, you must specify the full path to the file you want to send (including directories).

For instance "C:\Documents and Settings\X-Princess 51\My Videos\Sophie Moone and Sandy - lesbian scene.mpg".

If you don't specify the full path, it won't work.

To help you, there is a file selecting system. On the left you see the list of folders in blue. On the right you see the list of files in current folder. Above the folders list you can see current drive, up-directory button and hidden files showing checkbox.

When you click a directory, it automatically changes (to return to upper directory click "Go up").

Once you are in the right directory, select a file and the whole path shall appear in the path line. Then you can click "Send".

If the connection is successful and your pen-pal accepts the file, a file transfer begins.

Don't wonder why the file transfer speed is like dial-up and makes a large load on the CPU – the AES system on every block requires some CPU work, especially when the application is written in Java.

If your pen-pal wants to send file to you, then you first have a question whether you want to accept or decline the file transfer request. The question shows your pen-pal's nickname, file name and file size. If you decline the request, a clear code is sent to your pen-pal saying you decline the request.

Once you've accepted the request, you are asked where to save the file. You can choose the directory and change the file name in the path.

If the file exists you are asked if you wish to overwrite. If the size of the existing file is lower than the offered file, you have the resume option, meaning to complete the file starting from the next byte after the file size.

Once you've made your decision, the file transfer begins until the file is transferred.

File transfer – behind the curtains

The file transfer protocol is similar to the one of chat. I mean it has same blocks and same encryption.

Once you click "Send", a TCP connection is established with your pen-pal. Then your client send a "greeting block" to your pen-pal. The greeting block consists of two longs. The first long is a random number with one rule: it is always odd, meaning "file transfer". The second long means the size of the file you are going to transfer.

After sending the greeting block, your nickname and the file name are sent just like in chat.

Your pen-pal gets all the information and has choice to accept or decline. If he or she clicks on decline, a block with first long divisible by three (meaning "decline") and second long random is send.

If your pen-pal accepts the file then if he/she chooses to create a new file or to overwrite some existing file, then a block is sent to you where the remainder of dividing the first long by three is one and the second long is random number. If he or she chooses resume, then the first long gives remainder of two when divided by three and the second long means the position to start the transfer from. That position is of course one byte more than the file size he/she wants to resume.

Once the file transfer has begun, your client simply reads a block of sixteen bytes from your file, encrypts it and sends to the remote side.

To prevent buffer overload on the remote side, SDChat uses simple synchronization system. Every few blocks, your client pauses and waits for synchronization byte from your pen-pal (and your pen-pal knows to send that byte each few blocks). This is the only data which is not encrypted. The synchronization byte is only one byte and it is a completely random number. The number means nothing, but the existence of that byte means for your client to continue sending data.

Your pen-pals knows when the transfer is completed according to number of bytes received and the file size. The file size is not always divisible by the block size. Therefore, the unused bytes of the last block are random.

In case the connection is broken during the transfer, an incomplete file is saved and you can use the "resume" feature later.

In conclusion I'd say this application is very easy to use and contains no "fuckers-overfuckers" (backdoors and things like that). It works for you only. If you don't believe me then I can guarantee that by the fact SDChat is open-source. Check out the source code and that's it.

All you need to do is to run the application and enjoy high level of privacy. Just remember everything I've said here.

And questions – you may contact me and I'll be glad to help, However, if you're a complete lamer, don't expect me to teach you how to use a computer, sorry.

F.A.Q

Question:

What do you mean by saying "thoughts police"?

Answer:

Any sucker who can eavesdrop on your conversation. Could be private individuals, organizations, your boss, tax departments, private investigators, competitors, your insurance company, criminals or government. You may disagree with me, but eavesdropping on intimate conversation of other people without their agreement is a violation of basic human rights. That is why I use the term from George Orwell's book.

Question:

I'm a law abiding civilian and have nothing to hide, so why the heck do I need applications like this?

Answer:

First of all SDChat is for education purposes. Second, **everyone** has something to hide. It could be your intimate life, medical data, hobbies, financial status or anything else. Just like you probably wouldn't like foreigners watching you in the bathroom, there is other personal information you should keep secret. Third, imagine yourself you have a pen-pal of opposite sex, you and he/she like to run cyber-sex chats, but suddenly your country decides that cybersex is a heavy crime the penalty for which is life-sentence or capital punishment...

Question:

Will SDChat be always open-source and free?

Answer:

Yes. SDChat will never be commercial as I am a strong believer and fan of free and open-source software.

Question:

What can I do if I live in a country that violates basic human rights?

Answer:

SDChat does not have to be installed, it is a portable application. The only thing you need is JVM on your computer. You can download SDChat through proxy server, better use SSL. Then you can use SDChat to talk with your pen-pal and delete the application after the chat. Of course you shouldn't save the settings, you shall either use a complicated password or just memorize the key. Besides you would better not run SDChat under your operating system but rather from something like BartPE.

Question:

Why do you reveal the source code of SDChat?

Answer:

The main purpose of that is to guarantee you SDChat contains no backdoors or security flaws. The fact that SDChat is open-source can help users find bugs in the application, make individual changes for themselves or even send me improvement suggestions.

Question:

How can I be sure SDChat is compiled precisely from the published source code and no extra code was added to the Java intermediate code?

Answer:

You can simply recompile the source code and compare the results with the intermediate code. The only differences can be timestamps and some digital signatures. Besides an expert can analyze the code and make sure it contains no backdoors.