

Forum: [Hack Forums](#)

Thread: [Secure P2P Chat - Looking for testers](#)

HotNoob - January 9, 2015, 4:46 pm

[Video: <https://www.youtube.com/watch?v=OBhoo2mrwfU>]

Vid goes over basics, desc has some release notes to show where the current version is at.

instructions for installing JRE1.7 in linux is also in desc.

with v.06, after sending first message, wait for connection to complete until sending the 2nd messages, v.07 will have a que to remove the need for this.

Overall how it works,

client connects to server, performs 2x rsa handshakes, then continues to perform a dh handshake.

to connect, client sends a request to server, server forwards request to target client. client then returns to server if to connect or not, and provides the port and an accesscode and aes256 key to use to connect. server then forwards back to client. client then connects, sends access code, and encrypts the stream. then the clients perform 2x rsa, and 1x dh.

i have p2p group chats to work, but have not found any one to help me test that functionality. i have tested it between 2 connections, but need help with testing it between 3 or more people, and then also through various relays.

contact lists are stored on the server, but are encrypted by the client, so you have all of the comforts of normal chat systems without any security compromises. The local key used for encrypting the user data does not leave the clients computer, and neither is it saved anywhere on the computer, thus keeping it secure.

v.07 will have an updated ui
v.08 will have support for otp

later on i will be toughening up the security through some methods that i prefer not to disclose until i've programmed it into the client, as the concept could be revolutionary.

JIM NASIUM'S GYMNASIUM - January 9, 2015, 5:34 pm

Stopped watching after about 1:50. Listening to you badmouth Java because you need to download a runtime for the code to work was just cringeworthy.

JavaProphet - January 9, 2015, 5:52 pm

Throw it over TLS for identity verification and one extra layer of security.

D?rkStorm - January 9, 2015, 8:55 pm

You have included BouncyCastle, yet you complain about the JCE Unlimited Strength policy files? Were you just stapling together StackOverflow answers for your crypto? You can use BouncyCastle's API directly and skip all of the policy restrictions of Java's Security API.

```
Code PaddedBufferedBlockCipher cipher = new PaddedBufferedBlockCipher(new  
CBCBlockCipher(new AESEngine()));  
CipherParameters params = new ParametersWithIV(new KeyParameter(key), iv);  
cipher.init(false, params);
```

HotNoob - January 9, 2015, 9:07 pm

(Jan 9, 2015 09:55 PM)D?rkStorm Wrote: [\[CITE\]](#) You have included BouncyCastle, yet you complain about the JCE Unlimited Strength policy files? Were you just stapling together StackOverflow answers for your crypto? You can use BouncyCastle's API directly and skip all of the policy restrictions of Java's Security API.

```
Code PaddedBufferedBlockCipher cipher = new PaddedBufferedBlockCipher(new  
CBCBlockCipher(new AESEngine()));  
CipherParameters params = new ParametersWithIV(new KeyParameter(key), iv);  
cipher.init(false, params);
```

From BC website:

Quote:Note: with JDK 1.4 and later you will need to have installed the unrestricted policy files to take full advantage of the provider. If you do not install the policy files you are likely to get something like the following:

```
java.lang.SecurityException: Unsupported keysize or algorithm parameters  
    at javax.crypto.Cipher.init(DashoA6275)
```

The policy files can be found at the same place you downloaded the JDK.

<https://www.bouncycastle.org/specifications.html>

MrOrang - January 9, 2015, 11:46 pm

Uh seems cool but please explain why Java is terrible in any way. I'd like to know or are you a general Java basher without arguments.

HotNoob - January 10, 2015, 1:07 am

(Jan 10, 2015 12:46 AM)Looka Wrote: [\[CITE\]](#) Uh seems cool but please explain why Java is terrible in any way. I'd like to know or are you a general Java basher without arguments.

mainly the inconsistencies, and the massive lack of features... and the overall retardation as to how libraries are setup... and sheer stubbornness to add the much needed features.

like seriously... having signed bytes has gotta be one of the most retarded things ever.

nOrr - January 10, 2015, 1:47 am

(Jan 10, 2015 02:07 AM)HotNoob Wrote: [\[CITE\]](#) mainly the inconsistencies, and the massive lack of features... and the overall retardation as to how libraries are setup... and sheer stubbornness to add the much needed features.

like seriously... having signed bytes has gotta be one of the most retarded things ever.
Elaborate please. Also, there's nothing wrong with signed bytes.

D?rkStorm - January 10, 2015, 7:48 am

(Jan 9, 2015 10:07 PM)HotNoob Wrote: [\[CITE\]](#) From BC website:

<https://www.bouncycastle.org/specifications.html>

"...to take full advantage of the provider." = to use BouncyCastle with javax.crypto. Skip javax.crypto and use BC directly.

(Jan 10, 2015 02:07 AM)HotNoob Wrote: [\[CITE\]](#) mainly the inconsistencies, and the massive lack of features... and the overall retardation as to how libraries are setup... and sheer stubbornness to add

the much needed features.

like seriously... having signed bytes has gotta be one of the most retarded things ever.

Sounds more like you don't know the language well enough. Unlike Microsoft languages which give you MS' stack of software upfront (but lock you in), any feature API-wise you might be missing has most likely been implemented in some open source library for you to use. Even many of the core API parts have replacements (i.e. BouncyCastle can replace javax.crypto). The quantity of open source and commercial third party libraries is massive -- larger than any other language.

As for syntax simplicity and signed types: this is intentionally maintained for the purpose of familiarity: the rigidity and verbosity result in a fairly standardized method to code design so that software can be easily understood and maintained by any developer. This and excellent support for enterprise applications, among other things, is what makes it so extremely popular for business software.

HotNoob - January 13, 2015, 4:27 pm

Every other language in the world EXCEPT!!! for Java uses unsigned bytes...

And pretty much everything else uses unsigned bytes...

IP addresses, Hex, you name it. It's all 0 to 255.

Try doing a very simple check sum in java via adding up the values of the bytes...

You have to first convert the byte to and int, and then convert the int into it's positive value, AND THEN!!! add it to the sum.

vs, simply adding the value of the byte to the sum.

-Base- - January 13, 2015, 6:28 pm

(Jan 13, 2015 05:27 PM)HotNoob Wrote: [\[CITE\]](#) Every other language in the world EXCEPT!!! for Java uses unsigned bytes...

And pretty much everything else uses unsigned bytes...

Java likes to be different. What's wrong with being different? Huh, punk? Why don't you accept those who want to be different you sick bastard. ?

,
,
,
,
,
,
,
,
,
,
?

D?rkStorm - January 13, 2015, 7:42 pm

(Jan 13, 2015 05:27 PM)HotNoob Wrote: [\[CITE\]](#) Every other language in the world EXCEPT!!! for Java uses unsigned bytes...

And pretty much everything else uses unsigned bytes...

IP addresses, Hex, you name it. It's all 0 to 255.

Try doing a very simple check sum in java via adding up the values of the bytes...

You have to first convert the byte to and int, and then convert the int into it's positive value, AND

THEN!!! add it to the sum.

vs, simply adding the value of the byte to the sum.

Like many other things Java lacks, that is an intentional measure to prevent unwary developers from shooting themselves in the foot. Again, Java is not really meant to be a hobby language that is super fun to write in, it's meant to be an enterprise language where a couple hundred developers working on multiple parts of one huge project can work in a way that minimizes potential for careless mistakes.

HotNoob - January 13, 2015, 8:51 pm

The java devs who decided on using signed bytes, publicly stated that it was a huge fucking mistake to do so.

Plz, fan bois, pull your heads out of your asses.

nOrr - January 14, 2015, 2:22 am

(Jan 13, 2015 09:51 PM)HotNoob Wrote: [\[CITE\]](#) The java devs who decided on using signed bytes, publicly stated that it was a huge fucking mistake to do so.

Plz, fan bois, pull your heads out of your asses.

Honestly I haven't been in a situation where an unsigned byte would have saved my day, neither have I been in a situation where signed bytes have caused me serious issues that I could not work around easily. But then again I wouldn't really call myself a fan boi

JavaProfit - January 14, 2015, 5:26 am

(Jan 13, 2015 09:51 PM)HotNoob Wrote: [\[CITE\]](#) The java devs who decided on using signed bytes, publicly stated that it was a huge fucking mistake to do so.

Plz, fan bois, pull your heads out of your asses.

If all you're going to do is talk about how bad Java is, then just use something else. Honestly the other guys are right: signed bytes aren't that bad. Unsigned bytes make more sense in my opinion; it doesn't really make the language itself bad.