

# [NEWS] Defeating Lotus SameTime "Encryption"

**Source:** <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-08/0021.html>

---

**From:** SecuriTeam ([support\\_at\\_securiteam.com](mailto:support_at_securiteam.com))

**Date:** 08/10/03

To: [list@securiteam.com](mailto:list@securiteam.com)

Date: 10 Aug 2003 17:14:37 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.secureteam.com>

-- promotion

Get Thawte's New Step-by-Step SSL Guide for MSIS

In this guide you will find out how to test, purchase, install and use a Thawte Digital Certificate on your MSIS web server. Throughout, best practices for set-up are highlighted to help you ensure efficient ongoing management of your encryption keys and digital certificates. Get your copy of this new guide now: <http://ad.doubleclick.net/clk;5903126;8265119;j>

-----  
Defeating Lotus SameTime "Encryption"

---

## SUMMARY

Normal <http://www.lotus.com/home.nsf/welcome/sametime> Lotus SameTime login credential encryption with 1.5 and 3.0 Windows clients use RC2 (very improperly) to encrypt the password, and even send the key along with the login packet allowing an attacker to decrypt the credentials and steal a user's IM identity.

## DETAILS

Vulnerable systems:

- \* Lotus SameTime version 1.5
- \* Lotus SameTime version 3.0

Background:

Lotus SameTime is an Instant messaging protocol owned by Lotus Corporation, who in turn is owned by IBM. The Lotus SameTime web page says "... with over 8 million users, is the market leading instant messaging and Web conferencing solution for business." More market droid speak from

## Securiteam: [NEWS] Defeating Lotus SameTime "Encryption"

<<http://lotusdevelopmentadvisor.com/doc/11498>>

<http://lotusdevelopmentadvisor.com/doc/11498> says: "Because of questionable security and other shortcomings in consumer IM, companies have become increasingly concerned about unsanctioned use. Companies that realize the need for secure and reliable instant messaging turn from consumer IM to much more robust business IM platforms. For example, Lotus Sametime provides encryption, logging, archiving, directory integration, and integration into other business applications".

### Synopsis:

The following information details several severe flaws in the way encrypted logins and chats are handled. Users and administrators of SameTime should be aware of the vulnerabilities in the protocol.

In short, login messages contain the RC2/40 key in the login message itself. This allows an attacker to intercept and decrypt the user's password with very little effort. Additionally keys are transmitted with instant messages as well, and every instant message has 6 bytes of known-plaintext in the beginning of the data stream. Finally, the 10 byte RC2/40 keys are generated using only ASCII representations of decimal numbers 0-9 (hexadecimal 0x30 - 0x39), instead of using the full 256 possibilities available per each byte of the 10 byte key. This means there are only  $10^{10}$  possibilities for any SameTime key, rather than the potential  $256^{10}$ . Even a low-end (but fairly modern) personal computer can be used to brute force the key rather quickly. Then again, why would you need to since the key is right there in the login packet?

Users who think that they are being protected by SameTime "encryption" are not only risking having their passwords exposed, but also the messages they send which may contain confidential information (especially since SameTime is an IM aimed at corporate users). Additionally SameTime users should be aware that encryption is NOT end-to-end, and they can be snooped on by the server operator. There are several commercial products sold to do this, and they work regardless of the "encryption" selected by the client. For non-SEC use, this should be considered unacceptable.

### Login Message Analysis:

A Lotus SameTime 1.5 Login (extracted from tcpdump) message looks like this:

```
82 -- A sequence byte. - 0x81 was the first byte
00 -- Total data length
00 -- Total data length
00 -- Total data length
45 -- Total data length (69 bytes)
00 -- Message Type
01 -- Message Type
00 -- Options
00 -- Options
00 -- Channel ID
00 -- Channel ID
```

## Securiteam: [NEWS] Defeating Lotus SameTime "Encryption"

00 -- Channel ID  
00 -- Channel ID  
10 -- The type of login (Java / C++ / ActiveX etc..)  
02 -- The type of login (in this case 0x1002 == C++)  
00 -- Length of the following string  
11 -- Length of the following string (17 bytes)  
6a -- j  
6f -- o  
65 -- e  
62 -- b  
6C -- l  
6F -- o  
40 -- @  
97 -- a  
98 -- b  
2e -- .  
78 -- x  
79 -- y  
7a -- z  
2e -- .  
63 -- c  
6f -- o  
6d -- m  
00 -- length of opaque for auth data  
00 -- length of opaque for auth data  
00 -- length of opaque for auth data  
22 -- length of opaque for auth data (34 bytes)  
00 -- length of opaque for RC2 key  
00 -- length of opaque for RC2 key  
00 -- length of opaque for RC2 key  
0a -- length of opaque for RC2 key (10 bytes)  
33 -- opaque RC2 key data 1  
36 -- opaque RC2 key data 2  
30 -- opaque RC2 key data 3  
37 -- opaque RC2 key data 4  
34 -- opaque RC2 key data 5  
30 -- opaque RC2 key data 6  
33 -- opaque RC2 key data 7  
35 -- opaque RC2 key data 8  
30 -- opaque RC2 key data 9  
31 -- opaque RC2 key data 10  
00 -- length of opaque data for encrypted password  
00 -- length of opaque data for encrypted password  
00 -- length of opaque data for encrypted password  
10 -- length of opaque data for encrypted password (16 bytes)  
XX -- opaque password data 1 – data omitted  
XX -- opaque password data 2 – data omitted  
XX -- opaque password data 3 – data omitted  
XX -- opaque password data 4 – data omitted  
XX -- opaque password data 5 – data omitted  
XX -- opaque password data 6 – data omitted

## Securiteam: [NEWS] Defeating Lotus SameTime "Encryption"

XX --- opaque password data 7 - data omitted  
XX --- opaque password data 8 - data omitted  
XX --- opaque password data 9 - data omitted  
XX --- opaque password data 10 - data omitted  
XX --- opaque password data 11 - data omitted  
XX --- opaque password data 12 - data omitted  
XX --- opaque password data 13 - data omitted  
XX --- opaque password data 14 - data omitted  
XX --- opaque password data 15 - data omitted  
XX --- opaque password data 16 - data omitted  
00 --- Authentication Type  
02 --- Authentication Type

A 3.0 version of the session looks very much like this, but there is an extra 4 bytes which looper suspects is used in some way to try to partially address the weak key generation (but looper does not know for sure, since the 3.0 protocol isn't documented). Unfortunately the 3.0 client suffers from the same stupidity of having the key and their password sent along with the initial login message. Java SameTime API docs talk about the possibility of using 128bit RC2 with Diffie-Hellman key exchange. If the server is capable of doing this, why are the ubiquitous clients (both major Windows clients) doing logins this insecure way?

### The Details of the Aftermath:

Looper has noticed three serious flaws from the former analysis.

1. The RC2/40 key is right here in the same damn packet as the user's SameTime password that they key was used to encrypt. This reduces the encryption to nothing better than obfuscation on par with XOR with a known key.
2. Notice that the 10 bytes of the RC2 key are all in the range of 0x30 to 0x39 (ASCII for digits 0-9). This limits the possibilities to  $10^{10}$  or 10,000,000,000 rather than  $256^{10}$  or 1,208,925,819,614,629,174,706,176. As you can see, this drastically reduces the amount of time needed to brute force a key even if you happened to miss stealing it earlier.
3. The first 6 bytes of the encrypted password field are always the same. This makes it easy to use a known-plaintext attack to speed up the decryption process. A similar technique is used on encrypted message "channels" and there is some similar stupidity used there as well.

### ADDITIONAL INFORMATION

The information has been provided by <mailto:loper@hushmail.com> looper.

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

Securiteam: [NEWS] Defeating Lotus SameTime "Encryption"

=====  
=====

**DISCLAIMER:**

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.