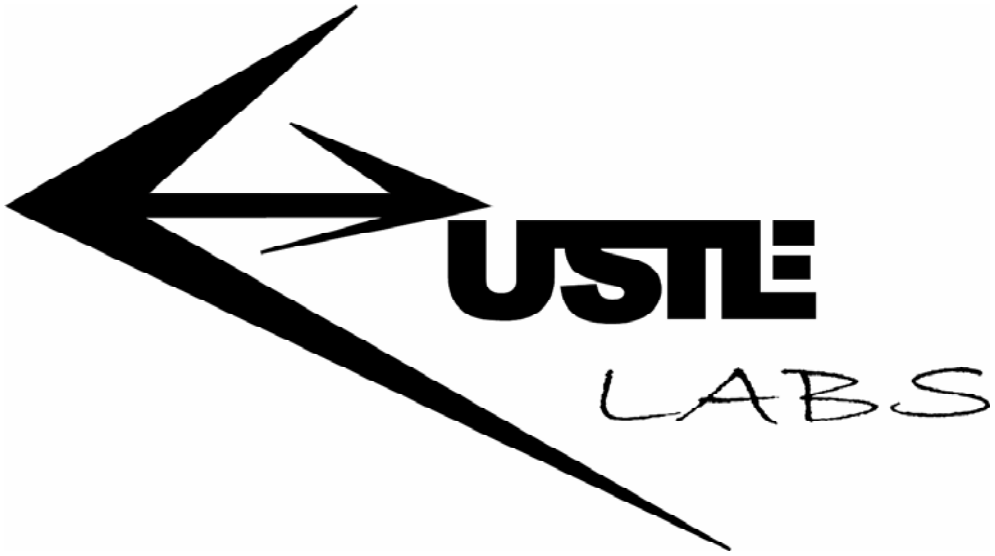

Product name	Shareaza
Advisory draft date	16 January 2006
Advisory status	Public Release
Researcher	Ryan Smith
E-Mail address	WhatsTheAddress@gmail.com

Summary

Product name	Shareaza
Versions affected	2.2.1.0 (Maybe earlier versions as well)
Vulnerability result	Remote code execution

Versions Affected

This report reflects the state of the code in the current (2.2.1.0) version. The author of this advisory did not believe the act of checking any other version would enhance this advisory. Therefore, it is uncertain whether versions prior to 2.2.1.0 are susceptible to the attacks described within this document.

Overview

There are several integer overflow vulnerabilities present within the Shareaza application. The developers of Shareaza have provided compatibility with a commendable number of file sharing networks. Within any file sharing scheme the user of the software is a part of, the Shareaza user may be enumerated, and the vulnerabilities may be exploited. As far as exploitation conditions are concerned, remote code execution is possible, although the conditions for exploitation are less than ideal.

Description

BTPacket.cpp

Within the BtPacket::ReadBuffer() routine, an effort is made to decode the packet's purported size. There is a check to ensure that the packet's length field is not larger than the size of the received buffer. This check will fail to validate that a value larger than 0xFFFFFFFB should not be parsed, due to an integer overflow, thus allowing the function to continue processing the packet.

EDPacket.cpp

Within the CEDPacket::ReadBuffer() routine, there are several blocks of validation code performed on a packet received from the network. There is a check to ensure that the packet's designated size field is less than the length of the packet that was received. This check is insufficient for any size larger than:

0xFFFFFFFF-sizeof(pHeader)+2

Thus, allowing the function to continue processing the packet.

Packet.h

The function CPacket::Write() contains code to copy data to a packet object. In order to handle expanding packets, while reducing performance degrading heap calls, the function will expand the packet by either 128 bytes, or the length of the element being added. There are two errors within this function that lead to vulnerabilities. The first error involves bypassing the heap expansion conditional by supplying a value large enough to wrap the integer. The second error involves continually expanding the size of the buffer until the size allocation wraps the integer.

Recommendation

Wait for a patched version.

Vendor Response

None.

Timeline

16 January 2006

Vendor notified of vulnerabilities and supplied with a preliminary draft of the vulnerabilities.

26 January 2006

Vendor did not have an official response; therefore the advisory was made public.