America Online Exploits Bug In Own Software

Some controversy developed in 1999 concerning the Instant Messaging service provided by America Online (AOL), which had at that time become very popular only recently. This article was updated in 2008, when moved to a new web site, but remains written as if the controversy is just breaking.

Background

The Instant Messaging service is available to anyone who registers a screen name and password at the AOL Instant Messenger (AIM) home page. Users are then to download and install the AIM client software, which AOL does not charge for. Instant Messaging is activated simply by running the AIM client while connected to the Internet. The AIM client uses the Internet to exchange packets of data with an AIM server at AOL. Many clients may be connected simultaneously with the one server. Instant Messaging as seen by AIM users is the result of the AIM clients and the AIM server interpreting the contents of their exchanged data packets according to a proprietary protocol.

Microsoft provides a similar Instant Messaging service through the Microsoft Network (MSN). The MSN Messenger Service has its own protocol for exchanges between MSN clients and an MSN server.

Users who run Microsoft's client for the MSN Messenger Service have some sort of access to AOL's Instant Messaging service, and hence to users who have registered with AOL but not with Microsoft. To achieve this, Microsoft has designed its MSN client with enough knowledge of the AIM protocol that an AIM server at AOL should not care whether it is talking with the intended AIM client software (as written by AOL) or with the MSN client software (as written by Microsoft).

Problem

In e-mail of dubious origin sent to security expert Richard M. Smith, it is alleged not only that the AIM client software has a so-called "buffer overflow" bug but also that AOL actually does use its knowledge of this bug to induce users' machines, which are running the AIM client software, to execute code that is downloaded from the AIM server. AOL is said to do this as a way for the AIM server to distinguish AIM clients from MSN clients so that the latter may be denied service.

Analysis

With the allegations is a reproduction of a data packet whose receipt is said to trigger the alleged "buffer overflow" bug. Inspection of how the AIM client software would handle this packet reveals a relevant coding weakness, specifically a buffer overflow in the client's handling of one particular case in the AIM protocol. For the purposes of this summary, this case may as well be named case 0013h, based on some identifying numbers, given as channel 02h, group 0001h, function 0013h in the accompanying bug details.

From a quick search of Google in January 2008, it seems that the AIM protocol, apparently now called OSCAR, is still not documented by AOL for public knowledge. Of course, it may be that I just didn't look hard enough for references. I am not a user of any sort of instant messaging, I have only limited interest in updating these pages, and I have no taste at all for hunting an obscurity in the face of numerous contrary indicators. There is seemingly no end of commentators who declare that OSCAR is not documented and who believe it never will be documented. Who am I to disagree? If it is documented, it is demonstrably less accessible than the unofficial documentation by Alexandr Shutko. What is here called case 0013h is there called Message Of The Day. Were official documentation available with sufficient ease, I would update these pages to use the official terminology. Presuming its absence, I stick with the numbers.

Bug Summary

When the AIM client receives a packet in case 0013h, the AIM client looks through the packet for data that the

protocol indicates is to be treated as a string. If this data is found in the packet, the AIM client makes a copy, which is then to be converted to a string by appending a null byte. To hold the copy and the appended null byte, the AIM client has a 0100h-byte buffer on the stack. The data to be copied is simply assumed to fit into the buffer, with room for the appended null byte. The AIM client is coded with no defence against the data being too long. If a packet in case 0013h is sent to the AIM client and this data that is to be converted to a string happens to be 0100h bytes or longer, then the AIM client will write to memory beyond the end of the buffer, corrupting whatever happens to be there. This is a classic buffer overflow bug.

Exploitation Summary

An ordinary, though certainly not necessary, effect of a program's corrupting memory on its stack is that the program crashes some time later. The particular packet presented in the e-mail to support the allegations against AOL fits case 0013h but contains 0118h bytes of string data. This is too long and will indeed induce the AIM client to corrupt memory, as described above. However, the AIM client does not crash.

The reason is that the packet data, as received from the AIM server, is contrived so that the corruption of memory by the AIM client is carefully controlled. The buggy routine in the AIM client is made to "return" to an address at which it is known there will be the bytes for a call esp instruction (actually provided in the bitmap for an icon in the AIM.EXE resources). The effect of this instruction is to start executing some of the packet data.

The next part of the contrivance is that this part of the packet data actually has been prepared as executable code. It does two things. One is to recover from the bug, so that the AIM client resumes apparently normal execution. The other, done as a little side-trip before recovery, is to form some of the downloaded packet data into a new packet that the AIM client is induced to send to the AIM server.

Interpretation

In effect, AOL has extended its AIM protocol retrospectively, adding a new subcase of case 0013h. The AIM client receives a packet that fits this subcase and it sends a packet back. The AIM server can use the response to test whether the AIM client "knows" the protocol extension.

Microsoft's client for the MSN Messenger Service does not know the protocol extension. Indeed, in the version inspected for this report (namely 1.0.0886), Microsoft's client ignores case 0013h altogether.

The protocol extension is essentially a test of genuineness. At the time of this test's introduction, no existing version of the MSN client can have known to respond to this new subcase of the AIM protocol. Indeed, no existing version of the AIM client was coded explicitly to respond to this subcase. The advantage of the trickery is that the protocol extension is known implicitly by any old AIM client that has the buffer overflow bug in its routine for handling case 0013h. Without requiring users to upgrade their AIM client software, AOL can have its AIM server test whether it is connected to a genuine AIM client—and no MSN client can pass this test until Microsoft builds knowledge of the protocol extension into a new version and gets users to upgrade.

This trickery must have been compelling to AOL's programmers and perhaps even to some managers. But it comes with two obvious disadvantages. One is that it requires AOL to withold its knowledge of a bug in its own software. Many software manufacturers, and very notably Microsoft, seem not to regard this sort of behaviour as even mildly unethical, let alone as shameful. Customers put up with it well enough, if only because of ignorance and impotence. A second disadvantage will trouble more people. The technique of AOL's contrivance involves getting the AIM client to execute code that the AIM server has supplied in the packet. Being able to download code to another machine and get it executed by that other machine is a hacker's dream. Software manufacturers who play this game have no credibility if ever they talk of caring for the security of their customers' computers.

Applicable Versions

The buffer overflow bug is in the PROTO.OCM module of the AIM client software. The versions studied and in which the bug has been identified are 2.0.912, 2.0.996, 2.1.1236 and 3.0.1415. File sizes and times are given in the following table.

Version	Size	Date
2.0.912	36,864	13th January 1999
2.0.996	36,864	25th February 1999
2.1.1236	36,864	18th June 1999
3.0.1415	39,424	17th August 1999

Existence of the buffer overflow bug in any given version of PROTO.OCM just means that users who have installed the AIM client software with this version of PROTO.OCM are exposed to having their machines execute code that has been downloaded to them as a data packet during an AIM session. It does not mean that any such exploitation has been observed to occur with the given PROTO.OCM version, just that the exploitation could be designed.

Research History

I began the analysis on 15th August with an overview of the downloaded AIM client package (version 2.1.1236). I located the buffer overflow bug in PROTO.OCM on 16th August by directing my inspection of the AIM client software to its treatment of the data packet given as evidence in the allegations. I did not register for Instant Messaging with AOL, nor connect to AOL for an AIM session. Indeed, I did not run the AIM client software beyond its setup, i.e., to extract the executables.

However, the corrupted return address as given in the allegations is wrong for exploiting the buffer overflow if received by this version of the AIM client. Though the bug was found in the code, the mechanism for exploiting it eluded me.

On 19th August, Andrew Schulman reported an AIM session in which he observed the AIM client (version 2.0.912) receive exactly the data packet given in the allegations. I registered with AOL as an AIM user and started debugging AIM sessions, hoping to observe an exploitation for myself. Unfortunately, there was no exploitation to observe: using version 2.1.1236, no data packet similar to that described in the allegations was received.

On 20th August, I returned to analysing the code, worked harder at it, and identified the mechanisms by which the AIM client might recover from the buffer overflow and be diverted to send data back to the AIM server. I then worked backwards to the mechanism for getting code in the packet to execute and was then able to tell Andrew Schulman where to find the bytes of the call esp instruction in his copy of AIM.EXE.

I downloaded version 3.0.1415 on 25th August 1999 after being alerted to a company press release dated 24th August 1999. (The link given is where this press release was published at the time, but seems to be invalid now.) Persistence of the buffer overflow bug in this “ALL NEW Version” was established immediately.

Acknowledgements

I am grateful to Andrew Schulman for directing me to this controversy and to both Andrew Schulman and Richard M. Smith for many helpful discussions during the analysis.

This page was created on 22nd August 1999 and was last modified on 7th November 2008.