

04-July-2006

Summary

WinRAR is an archive manager that supports a diverse range of formats. This list of formats includes RAR, ZIP, CAB, ARJ, LZH, TAR, GZ, BZ2, ACE, UUE, JAR, ISO, 7Zip, and Z.

While processing LHA files, WinRAR concatenates the directory-name and the filename portions of an archive. Due to a lack of constraints while copying data, two stack-buffer overflows can result.

Impact

These vulnerabilities are present by default in WinRAR. An attacker would need to convince a WinRAR user to open a specially crafted file. This file can have any extension as long as WinRAR is configured to process it. Successful exploitation of these vulnerabilities results in code execution with the full privileges of the current user. Since these exploits are stack based, and due to specific code constructs, exploitation can be made reliable.

Affected software

WinRAR – At least versions less-than 3.60 beta 7 and greater-than 3.0, although others may be affected as well

Credit

These vulnerabilities were researched by Ryan Smith.

Contact

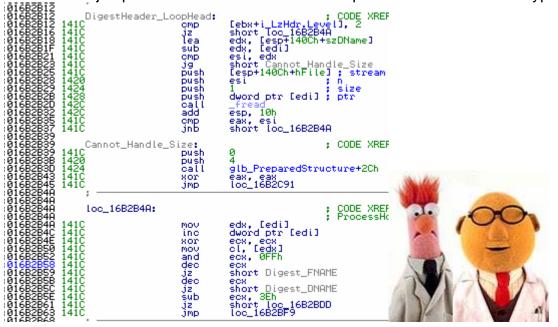
advisories@hustlelabs.com





Details

The following code processes LHA extended-headers. For level-1 LHA headers, the code reads in extra data from a file. The code then executes a switch statement to jump to a location that handles the specific extended-header type.



Here the code reads in extended-headers of the filename type. The code reads filename sizes of up-to 0xFC bytes and stores this user-supplied data into the szFilename buffer.

:016B2B68 Digest_FNAME: :016B2B68 141C :016B2B68 141C :016B2B69 141C :016B2B70 141C :016B2B75	cmp jl mov	esi, 100h ; CODE XREF: ProcessHdr+2E1⊫j short loc_16B2B75 esi, 0FFh
01682875 loc_1682875: 01682875 141C 01682877 141C 01682879 :	xor jmp	; CODE XREF: ProcessHdr+2F6 j eax, eax short loc_16B2B87
01682879 01682879 01682879 141C 01682879 141C 01682870 141C 01682880 141C 01682880 141C 01682887 141C 14	Mov and inc inc lea cmp jmp jmp	; CODE XREF: ProcessHdr+314[j edx, [edx] cl, 0FFh Lebx+eax+i_LzHdr.szFileName], cl dword ptr [edi] eax ; CODE XREF: ProcessHdr+2FF1j eax, edx short loc_1682879 byte ptr Lebx+esi+(i_LzHdr.DTSLastModified+3)], 0 short try_next_hdr

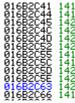




The code in the following image is responsible for processing LHA extendedheaders of the directory name type. The code will copy a user-supplied buffer up to 0x3FC bytes in length into the szDName variable.

01682895 01682895 01682895 01682895 01682898 01682898 01682890 01682890	Digest_DNAME: 141C 141C 141C	çmp jl mov	; CODE XREF: Process short loc_16B2BA2 esi, 3FFh
016B2BA2 016B2BA2 016B2BA2 016B2BA4 016B2BA6	loc_16B2BA2: 141C	Xor JMP	; CODE XREF: Process eax, eax short loc_16B2BB7
016828A6 016828A6 016828A6 016828A8 016828A8 016828AA 016828B4 016828B4 01682886 01682886 01682886	141C	mov mov and mov inc inc	; CODE XREF: Process cl, [edi] cl, 0FFh [esp+eax+140Ch+szDName], cl dword ptr [edi] eax
01682888 01682888 01682888 01682888 01682888 01682888 01682882 01682882 01682882 016828282 016828282 01682804 016804 0168040000000000000000000000000000000000	1410 1410 1410 1410	lea cmp jl lea mov call add mov jmp	; CODE XREF: Process eax, edx short loc_1682BA6 [esptesit140Ch+ucaFileData+0FFCh], 0 eax, [esp+140Ch+szDName]; string dl, ', ' value ReplaceChars esi, 0FFFFFFDh [espt140Ch+SzDName], esi short try_next_hdr

The next image is the vulnerable portion of the code. The program takes the two user-supplied values, one up to 0x3FC bytes in length, the other up to 0xFC bytes in length, and concatenates them into a buffer that is 0x400 bytes in length. The code then copies the resultant buffer, up to 0x4F8 bytes in length, to the szFileName buffer that is only 0xFF bytes in length. This buffer mismanagement results in two stack based overflows.



lea push lea push call add Īēi

esi, [ebx∓i_LzHdr.szFileName] esi src eax, [esp+1410h+szDName] eax 8 [esp+140Ch+szDName] src edx, edx dest esi esp, 8 ebp, [esp+140Ch+SzDName]







Remediation

The code should either truncate the strings, or allocate more space for the strings.

Version 3.6 Beta 7 corrects the issue mentioned in this document. This version should be installed in order to mitigate the vulnerability. If the old version must be used, it may be possible to copy the lzh.fmt file from the new installer into the current directory. As well, if LHA compression is not needed, the file lzh.fmt may be removed from the installation directory.



Timeline of Events

04-July-2006 – Advisory draft date 11-July-2006 – Vendor notification 12-July-2006 – Vendor created a patch 13-July-2006 – Vendor released patched version 18-July-2006 – Advisory made public





Attributions

The images of The Muppet Show's Beaker and Dr. Bunsen were taken from http://www.getbert.com, http://www.forskning.no and http://newsimg.bbc.co.uk.

Code and cross-reference screenshots captured using IDA (http://www.datarescue.com).

Flawed code obtained from RARLabs (http://www.rarlab.com).

The Creative Commons license-notification image borrowed from http://www.creativecommons.org.

License

This work is licensed under the Creative Commons Attribution 2.5 License. To view a copy of this license, visit http://creativecommons.org/licenses/by/2.5/ or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.

Attribution should be provided both in the form of a link or reference to http://www.hustlelabs.com and a copy of the researchers' names listed under the *Credit* section of this document.

All other trademarks and copyrights referenced in this document are the property of their respective owners.

