Tumbleweed Email Firewall
Remote Stack Overflow

04-July-2006

Summary

Tumbleweed's Email Firewall (EMF) blocks spam and viruses, phishing and email fraud, and keeps hackers from compromising your network. To ensure compliance with government and industry regulations like HIPAA, GLBA, Sarbanes-Oxley and Safe Harbor (EU), MailGate Email Firewall provides sophisticated filtering, monitoring, encryption and reporting capabilities. According to product literature, Tumbleweed is used by: over 150 healthcare providers, the Department of Defense, the Department of Homeland Security, all four branches of the US Military, state and local governments internationally, 8 of the top 10 US banks, 4 of the top 5 Canadian banks, and 6 of the top 10 European banks.

Tumbleweed's EMF Decomposer, a component that decompresses incoming e-mail attachments, has three separate vulnerabilities within its LHA processing routines. The first issue causes the LHA processing engine to exhibit a stack-buffer overflow while processing extended-header filenames. The second issue is a stack overflow while processing LHA extended-header directory names. The third issue is a buffer overflow during a sprintf call while processing long filenames contained in an LHA archive.

Impact

These vulnerabilities are present by default in Tumbleweed's Email Firewall. To exploit these vulnerabilities, an attacker only needs to send an e-mail to an organization running Tumbleweed; it is not necessary that the e-mail is opened. Successful exploitation of these vulnerabilities results in remote code execution with the full privileges of the MMSDecompose process. The default settings allow an attacker to obtain super-user access to the machine. Since these vulnerabilities are stack based overflows, exploits can be made to work reliably.

Affected software

Tumbleweed Email Firewall (All Versions)

Credit

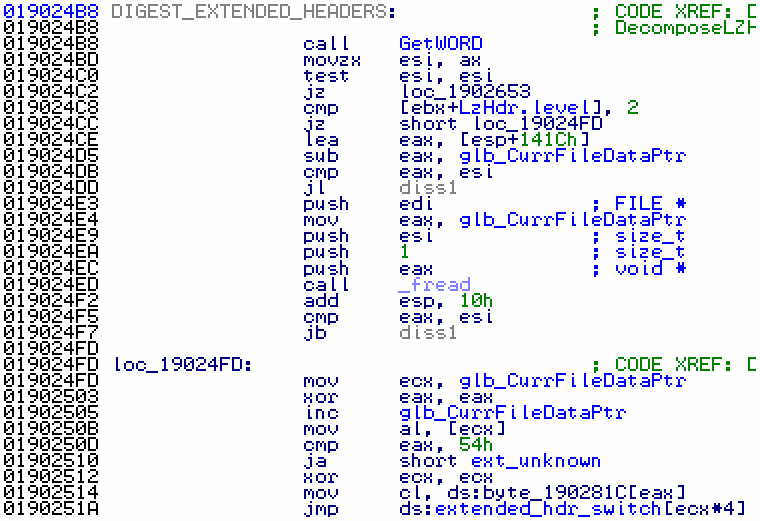
These vulnerabilities were researched by Ryan Smith.

Contact

advisories@hustlelabs.com

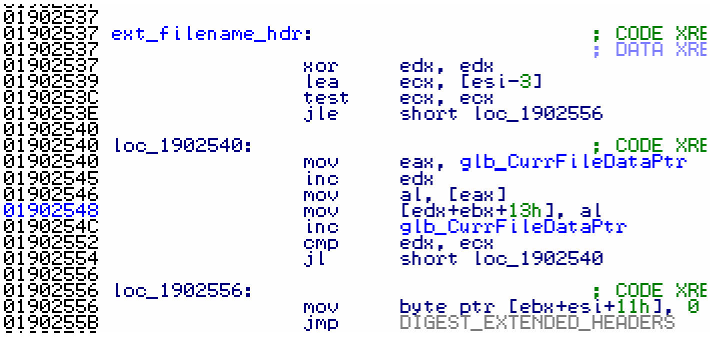
Details

In the following code segment, the program reads a word-sized value from the file: the LHA extended-header size. If the LHA header level is equal-to 1 then the program will read in more data to a buffer. Next, depending on the value of the extended-header-type byte, the program branches to an area of code that handles the specific type or a generic handler if it’s an unrecognized type.



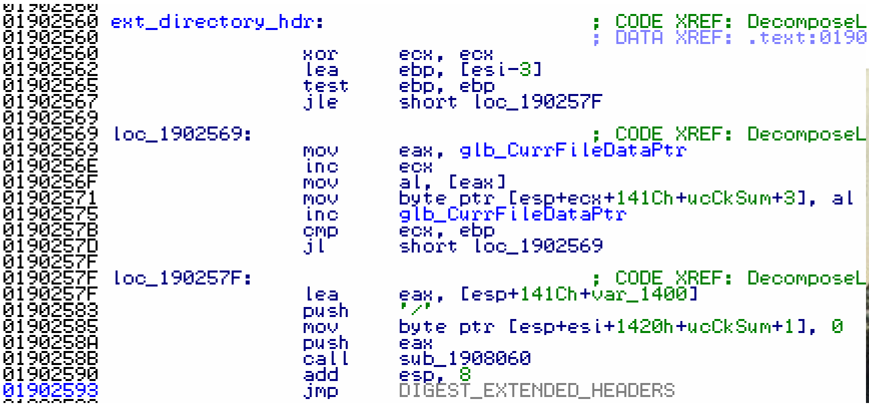
```
019024B8 DIGEST_EXTENDED_HEADERS:                        ; CODE XREF: [
019024B8                                                 ; DecomposeLZH
019024B8                 call    GetWORD
019024BD                 movzx   esi, ax
019024C0                 test    esi, esi
019024C2                 jz      loc_1902653
019024C8                 cmp     [ebx+LzHdr.level], 2
019024CC                 jz      short loc_19024FD
019024CE                 lea     eax, [esp+141Ch]
019024D5                 sub     eax, glb_CurrFileDataPtr
019024DB                 cmp     eax, esi
019024DD                 jl      diss1
019024E3                 push    edi             ; FILE *
019024E4                 mov     eax, glb_CurrFileDataPtr
019024E9                 push    esi             ; size_t
019024EA                 push    1               ; size_t
019024EC                 push    eax             ; void *
019024ED                 call    _fread
019024F2                 add     esp, 10h
019024F5                 cmp     eax, esi
019024F7                 jb      diss1
019024FD
019024FD loc_19024FD:                            ; CODE XREF: [
019024FD                 mov     ecx, glb_CurrFileDataPtr
01902503                 xor     eax, eax
01902505                 inc     glb_CurrFileDataPtr
0190250B                 mov     al, [ecx]
0190250D                 cmp     eax, 54h
01902510                 ja      short ext_unknown
01902512                 xor     ecx, ecx
01902514                 mov     cl, ds:byte_190281C[eax]
0190251A                 jmp     ds:extended_hdr_switch[ecx*4]
```

If the extended-header-type byte is equal to 0x01, then the following code parses the data for the header. The size allocated for this buffer is equal-to 0x100 bytes, but there is no length restriction. Thus, an attacker can supply a value greater-than 0x100 in an archive file to cause a buffer-overflow.



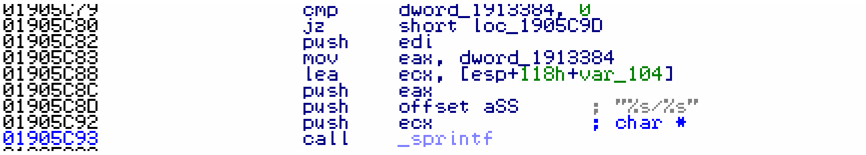
```
01902537
01902537 ext_filename_hdr:                       ; CODE XRE
01902537                                         ; DATA XRE
01902537                 xor     edx, edx
01902539                 lea     ecx, [esi-3]
0190253C                 test    ecx, ecx
0190253E                 jle     short loc_1902556
01902540
01902540 loc_1902540:                            ; CODE XRE
01902540                 mov     eax, glb_CurrFileDataPtr
01902545                 inc     edx
01902546                 mov     al, [eax]
01902548                 mov     [edx+ebx+13h], al
0190254C                 inc     glb_CurrFileDataPtr
01902552                 cmp     edx, ecx
01902554                 jl      short loc_1902540
01902556
01902556 loc_1902556:                            ; CODE XRE
01902556                 mov     byte ptr [ebx+esi+11h], 0
0190255B                 jmp     DIGEST_EXTENDED_HEADERS
```

As well, if the extended-header-type byte is equal to 0x02, then this next piece of code parses the data for the header. The size allocated for this buffer is equal-to 0x100 bytes; however, there is no restriction for the number of bytes the program will copy from the file, to the buffer. An attacker can supply a value greater-than 0x100 to cause a buffer-overflow.



```
01902560 ext_directory_hdr:                                     ; CODE XREF: DecomposeL
01902560                                                        ; DATA XREF: .text:0190
01902560                         xor     ecx, ecx
01902562                         lea     ebp, [esi-3]
01902565                         test    ebp, ebp
01902567                         jle     short loc_190257F
01902569
01902569 loc_1902569:                                           ; CODE XREF: DecomposeL
01902569                         mov     eax, glb_CurrFileDataPtr
0190256E                         inc     ecx
0190256F                         mov     al, [eax]
01902571                         mov     byte ptr [esp+ecx+141Ch+ucCkSum+3], al
01902575                         inc     glb_CurrFileDataPtr
0190257B                         cmp     ecx, ebp
0190257D                         jl      short loc_1902569
0190257F
0190257F loc_190257F:                                           ; CODE XREF: DecomposeL
0190257F                         lea     eax, [esp+141Ch+var_1400]
01902583                         push    '/'
01902585                         mov     byte ptr [esp+esi+1420h+ucCkSum+1], 0
0190258A                         push    eax
0190258B                         call    sub_1908060
01902590                         add     esp, 8
01902593                         jmp     DIGEST_EXTENDED_HEADERS
```

The next code excerpt is responsible for concatenating a temporary directory and the filename strings, in order to derive the path for the output of decompression. The first variable to the sprintf call is the format string “%s/%s”, the second variable is the temporary pathname, and the third is the filename. Of these parameters, the filename parameter is a user controlled value whose size should be no larger than 0x100 bytes (Including a null-termination character), according to the program’s data structures. Ignoring the fact that the size restriction is not enforced, a temporary pathname greater-than 2 characters in length will allow a buffer overflow to occur even for archive names that are legitimately sized.



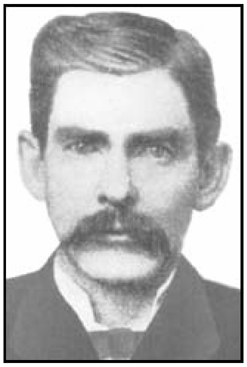
```
01905C79                         cmp     dword_1913384, 0
01905C80                         jz      short loc_1905C9D
01905C82                         push    edi
01905C83                         mov     eax, dword_1913384
01905C88                         lea     ecx, [esp+118h+var_104]
01905C8C                         push    eax
01905C8D                         push    offset aSS           ; "%s/%s"
01905C92                         push    ecx                  ; char *
01905C93                         call    _sprintf
```

Remediation

The code should be modified such that there is a standard maximum length of a path. If the path exceeds the maximum length, then the file should be rejected, or the path truncated.

Though Tumbleweed won't release a patch, they officially recommend the following actions:

- Stopping the EMF services
- Removing or renaming the wlha32.dll file, found in the EMF install directory.
- Restarting the EMF services



Timeline of Events

04-July-2006 – Advisory draft
11-July-2006 – Vendor notification
24-July-2006 – Vendor released customer notification and the patch

Attributions

Images of Billy the Kid, Jesse James, Butch Cassidy, and Doc Holiday were taken from Wikipedia.
(http://www.wikipedia.org)

Image of the tumbleweed along a road was photographed by AV Smith, from the Galveston Arts Center. (http://www.galvestonartscenter.org)

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

Flawed code and marketing information obtained from Tumbleweed (http://www.tumbleweed.com).

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

License

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.