* + 1. Định dạng chữ ký của ClamAV
       1. MD5

Cách dễ nhất để tạo chữ ký cho ClamAV là sử dụng mã checksum MD5, tuy

nhiên phương pháp này chỉ có thể sử dụng cho mã độc tĩnh. Tạo mã MD5: *zolw@localhost:/tmp/test$ sigtool --md5 test.exe > test.hdb zolw@localhost:/tmp/test$ cat test.hdb 48c4533230e1ae1c118c741c0db19dfb:17387:test.exe*

Sử dụng chữ ký:

*zolw@localhost:/tmp/test$ clamscan -d test.hdb test.exe test.exe: test.exe FOUND*

*----------- SCAN SUMMARY -----------Known viruses: 1*

*Scanned directories: 0*

*Engine version: 0.92.1*

*Scanned files: 1*

*Infected files: 1*

*Data scanned: 0.02 MB Time: 0.024 sec (0 m 0 s)*

* + - 1. Tạo MD5 cho một phần PE file

ClamAv hỗ trợ việc tạo chữ ký MD5 cho một phần cụ thể của PE file. Chữ ký này được lưu trữ trong file mdb với định dạng sau:

*PESectionSize:MD5:MalwareName*

Cách dễ dàng nhất để tạo chữ ký MD5 cho một phần PE file là trích xuất phần đó ra một file riêng biệt và sau đó chạy với tùy chọn -mdb với ClamAV.

* + - 1. Chữ ký đựa trên phần thân chương trình

Clam AV lưu trữ tất cả các chữ ký đựa trên phần thân của file trong một định dạng hệ 16 (hexa). Một chữ ký dạng hexa nghĩa là phần thân của file được chuyển đổi sang dạng chuỗi hệ 16 và có thể mở rộng khi sử dụng các ký tự khác nhau.

* + - * 1. Định dạng hexa

Sử dụng công cụ sigtool –hex-dump để chuyển dữ liệu bất kỳ sang dạng chuỗi hexa

*zolw@localhost:/tmp/test$ sigtool --hex-dump How do I look in hex?*

*486f7720646f2049206c6f6f6b20696e206865783f0a*

* + - * 1. Ký tự đại diện

ClamAv hỗ trợ các phần mở rộng dưới đây cho chữ ký hex:

?? : Phù hợp với bất kỳ byte nào

a?: Phù hợp với một nibble cao (bốn bít cao)

?a: phù hợp với nibble thấp ( bốn bít thấp)

\*: Phù hợp với bất kỳ số lượng byte

{}: Phù hợp với n byte

{-n}: phù hợp với nhỏ hơn hoặc bằng n byte

{n-}: phù hợp với lớn hơn hoặc bằng n byte

{n-m}: phù hợp n-m byte với n>m

(aa||bb||cc||…): phù hợp với aa hoặc bb hoặc cc ...

! (aa||bb||cc||…): phù hợp với bất kỳ byte nào trừ aa, bb,cc,…

HEXSIG[x-y] aa or aa[x-y]HEXSIG : phù hợp với aa gắn vào một chữ ký hex

(B): phù hợp với từ (bao gồm tập tin giới hạn)

(L): phù hợp với CR, CRLF hoặc tập tin giới hạn

Chữ ký phạm vi \* và {} hầu như tách một hex thành hai phần.

* + - * 1. Định dạng chữ ký cơ bản

Ví dụ về định dạng chữ ký :

*MalwareName=HexSignature*

ClamAv sẽ quét tất cả tập tin để tìm chữ ký hex. Tất cả chữ ký kiểu này được lưu dưới dạng file .db.

* + - * 1. Định dạng chữ ký mở rộng

Định dạng chữ ký mở rộng cho phép bổ sung thông tin đặc điểm kỹ thuật như một loại tập tin mục tiêu, offset của virus hoặc phiên bản công cụ, giúp phát hiện tốt hơn. Định dạng như sau:

*MalwareName:TargetType:Offset:HexSignature[:MinFL:[MaxFL]]*

TargetType có giá trị là một trong các số sau đại diện cho kiểu file cụ thế:

0: bất kỳ file nào

1:file thực thi di động đối với cả 32 bit và 64 bit

2: file nằm trong OLE2: file ảnh, VBscript …

3: HTML

4: Tập tin mail

5: đồ họa

6: ELF

7: file text mã ASCII

8: không sử dụng

9: File Mach – O

Offset là một dấu hoặc số thập phân n có thể kết hợp với việc thay đổi đặc

biệt:

\* : bất kỳ

n:offset tuyệt đối

EOF – n: kết thúc file trừ đi n byte

Chữ ký cho PE,ELF, Mach-O file hỗ trợ:

EP + n:entry point cộng n byte

EP – n: entry point trừ n byte

Sx + n: bắt đầu từ section x của dữ liệu cộng n byte

Sx – n: bắt đầu từ section x của dữ liệu trừ n byte

SL + n: kết thúc của section của dữ liệu cộng n byte

SL – n: kết thúc của section của dữ liệu trừ n byte

* + - 1. Chữ ký biểu tượng cho PE file

ClamAV 0.96 có một khoảng biểu tượng mờ để giúp phát hiện các tập tin thực thi mã độc đã ngụy trang bằng cách tìm kiếm các tập tin hình ảnh, office, văn bản, hay tương tự thế.

Biểu tượng phù hợp chỉ được thông qua chữ ký .ldb bằng cách sử dụng thuộc tính đặc biệt thẻ IconGroup1 hoặc IconGroup2. Định danh hai nhóm các biểu tượng được lưu trong tập tin cơ sở dữ liệu .ldb. Định dạng của tập tin .ldb là :

*ICONNAME:GROUP1:GROUP2:ICON\_HASH*

* ICON\_NAME llà một chuỗi định dannh duy nhất cho biểu tượng cụ thể
* Group1 là một chuỗi từ định danh nhóm đầu tiên của các biểu tượng (Icongroup1)
* GROUP2 là một chuỗi định danh nhóm thứ hai của các biểu tượng (IconGroup2)
* ICO\_HASH là một mã băm mò của hình ảnh biểu tượng
  + - 1. Chữ ký cho các siêu dữ liệu thông tin phiên bản trong PE file

Từ phiên bản CLamAV 0.96, nó đã dễ dàng kết hợp một số thông tin xây dựng vào các PE file (gồm file thực thi và thư viện lien kết động). Bất kỳ khi nào tra cứu thuộc tính của một tập tin thực thi PE trong windows, chúng ta sẽ thấy một loạt các chi tiết đó.

Những thông tin được lưu trữ trong một khu vực đặc biệt của nguồn tập tin kèm theo tên của VS\_VERSION\_INFOMATION (hoặc versioninfo). Nó được chia làm hai phần. Phần đầu tiên là một loạt cá số và cờ cho ta thấy phiên bản tập tin. Lúc đầu nó được sử dụng với các trình cài đặt sau khi phân tích nó sẽ có thể xác định xem việc thực thi hoặc thư viện nhất định sẽ được nâng cấp, ghi đè hoặc phát triển lên. Phương pháp này không bao giờ thực sự được sử dụng.

Phần thứ hai là một danh sách đơn giản của khóa, chuỗi giá trị.Nó dành cho thông tin người dùng và hoàn toàn bỏ qua bởi hệ diều hành. Ví dụ khi nhìn vào ping.exe ta sẽ thấy công ty là “Microsoft Corpo-ration”, phần mô tả là “TCP/IP

Ping command”,tên nội bộ là “ping.exe”…. Tùy thuộc vào phiên bản hệ điều hành, một số khóa sẽ có thể hiển thị thông tin đặc biệt trong hộp thoại thuộc tính, tuy tên nội bộ là như nhau.

Để phù hợp với một cặp khóa/giá trị thông tin phiên bản các offset đặc biệt của file gắn VI đã được đưa ra. Điều này tương tự như cá kiểu gắn khác ngoài trừ việc thay đổi để phù hợp với mẫu hex lặp lại của một offset riêng biệt, nó sẽ kiểm tra chính nó và mỗi cặp khóa/giá trị trong tập tin.Thẻ VI không cần và cũng không chấp nhận một +/- offset giống như các ví dụ EP +1. Đối với các chữ ký hex của chính nó, nó chỉ là UTF 16 kết xuất của khóa và giá trị. Chỉ ký tự đại diện ?? và (aa||bb) được cho phép trong chữ ký.Thông thường chúng ta không cần phải bận tâm tìm ra đáp án: bởi vì mỗi cặp khóa/giá trị cung với chữ ký VI dựa trên tương ứng được in bằng clamscan khi dùng tùy chọn –debug.

Ví dụ: *clamscan --debug freecell.exe*

*[...]*

*Recognized MS-EXE/DLL file in cli\_peheader*

*versioninfo\_cb: type: 10, name: 1, lang: 410, rva: 9608 cli\_peheader: parsing version info @ rva 9608 (1/1)*

*VersionInfo (d2de): ’CompanyName’=’Microsoft Corporation’ - VI:43006f006d00700061006e0079004e0061006d006500000000004d006900*

*630072006f0073006f0066007400200043006f00720070006f0072006100740*

*069006f006e000000*

*VersionInfo (d32a): ’FileDescription’=’Entertainment Pack*

*FreeCell Game’ - VI:460069006c006500440065007300630072006900700 0740069006f006e000000000045006e007400650072007400610069006e006d*

*0065006e00740020005000610063006b0020004600720065006500430065006 c006c002000470061006d0065000000*

*VersionInfo (d396): ’FileVersion’=’5.1.2600.0 (xpclient.010817*

*-1148)’ - VI:460069006c006500560065007200730069006f006e00000000 0035002e0031002e0032003600300030002e003000200028007800700063006 c00690065006e0074002e003000310030003800310037002d00310031003400 380029000000*

*VersionInfo (d3fa): ’InternalName’=’freecell’ - VI:49006e007400 650072006e0061006c004e0061006d006500000066007200650065006300650*

*06c006c000000*

*VersionInfo (d4ba): ’OriginalFilename’=’freecell’ - VI:4f007200 6900670069006e0061006c00460069006c0065006e0061006d0065000000660*

*0720065006500630065006c006c000000*

*VersionInfo (d4f6): ’ProductName’=’Sistema operativo Microsoft*

*Windows’ - VI:500072006f0064007500630074004e0061006d00650000000 000530069007300740065006d00610020006f00700065007200610074006900*

*76006f0020004d006900630072006f0073006f0066007400ae0020005700690*

*06e0064006f0077007300ae000000*

*VersionInfo (d562): ’ProductVersion’=’5.1.2600.0’ - VI:50007200 6f006400750063007400560065007200730069006f006e00000035002e00310*

*02e0032003600300030002e0030000000 [...]*

Mặc dù chữ ký VI ở trên được sử dụng trong chữ ký logic nhưng ta vẫn có thể thử dung chúng bình thường trong file .ndb:

*my\_test\_vi\_sig:1:VI:paste\_your\_hex\_sig\_here*

Nếu muốn giải mã một chữ ký VI cơ bản ta dùng :

*echo hex\_string | xxd -r -p | strings –el*

* + - 1. Chữ ký dựa trên siêu dữ liệu

ClamAV 0.96 cho phép tạo chữ ký chung phù hợp với các tập tin được lưu

trữ trong các kiểu chứa khác nhau đáp ứng các điều kiện cụ thể. Định dạng như sau: *VirusName:ContainerType:ContainerSize:FileNameREGEX: FileSizeInContainer:FileSizeReal:IsEncrypted:FilePos: Res1:Res2[:MinFL[:MaxFL]]*

* VirusName: Tên của virus hiển thị khi khớp chữ ký
* ContainerType gồm các định dạng CL\_TYPE\_ZIP, CL\_TYPE\_RAR, CL\_TYPE\_ARJ,CL\_TYPE\_CAB,CL\_TYPE\_7Z, CL\_TYPE\_MAIL,CL\_TYPE\_(POSIX|OLD)\_TAR, CL\_TYPE\_CPIO\_(OLD|ODC|NEWC|CRC) hoặc \*
* ContainerSize: Kích thước nơi chứa tập tin
* FileNameREGEX: Mô tả tên file đích
* FileSizeInContainer: thường được nén kích thước,
* FileSizeReal: thường không nẽ kích thước, MAIL,TAR và CPIO thì

ngược lại

* IsEncrypted: 1nếu file mã hóa, 0, nếu không mã hoá, \* để bỏ qua
* FilePoss: Vị trí file trong nới chứa
* Res1: khi ContainerType là CL\_TYPE\_ZIP or CL\_TYPE\_RAR
* Res2: không sử dụng với ClamAV0.96
* Chữ ký được lưu trữ trong file .cdb.
  + - 1. Chữ ký trên siêu dữ liệu ZIP/RAR

Nó chỉ được sử dụng lưu trữ chữ ký đối với các định dạng Zip hoặc Rar và có

định dạng:

*virname:encrypted:filename:normal size:csize:crc32:cmethod: fileno:max depth*

* Virname: Tên virus
* Encrypted: cờ mã hóa, 1- mã hóa , 0 – không mã hóa
* Filename: Tên file
* Normal size : kích thước file không nén (\* để bỏ qua)
* Compressed size: kích thước file khi nén(\* để bỏ qua)
* CRC32(\* để bỏ qua)
* Cmethod: phương thức nén (\* để bỏ qua)
* Fileno: vị trí file trong kho lưu trữ (\* để bỏ qua)
* Max depthSố lượng tối đa dữ liệu lưu trữ chồng nhau(\* để bỏ qua)

Cơ sở dữ liệu này được xuất dưới dạng file .zmd hoặc .rmd

* + - 1. Danh sách trắng

Danh sách trắng là một tập tin cụ thể sử dụng chữ ký MD5 và đặt trong một tập tin cớ sở dữ liệu có định dạng .fb

Để danh sách một chữ ký cụ thể từ cơ sở dữ liệu ta chỉ cần thêm tên của nó vào một tập tin cục bộ gọi là local.ign2 được lưu trữ trong thư mục cơ sở dữ liệu. Ta cũng có thể thực hiện theo tên chữ ký với MD5 của phần nhập toàn bộ cơ sở dữ liệu cho chữ ký .Ví dụ:

*Eicar-Test-Signature:bc356bae4c42f19a3de16e333ba3569c*

* + - 1. Danh sách chữ ký theo tên

ClamAv sử dụng các tiền tố dưới đây làm tên chữ ký:

* Worm: Sâu mạng
* Trojan: Chương trình cửa hậu
* Adware: adware
* Flooder: flooder
* HTML: tập tin HTML
* Email: Thư điện tử
* IRC: trojan IRC
* JS: Mã độc Java Script
* PHP: Mã độc PHP
* ASP: Mã độc ASP
* VBS: Mã độc VBS
* BAT: Mã độc BAT
* W97M,W2000M:virus macro trên word
* X97M, X2000M: virus macro trên excel
* O97M,O200M: virus macro chung trên office
* DoS: phân mềm tấn công từ chối dịch vụ
* DOS: mã độc DOS trước đây
* Exploit: lỗ hổng bảo mật phổ biến
* VirTool: Công cụ tạo virus
* Dialer: dialer
* Joke: lừa đảo

Quy tắc đặc tên của ClamAV:

* Luôn sử dụng một hậu tố zippwd vào tên mã độc hại cho loại ZMD
* Luôn sử dụng một hậu tố rarpwd vào tên mã độc hại cho loại RMD
* Chỉ sự dụng các ký tự chữ và số, dấu gạch ngang “-“, dấu chấm “.”, dấu gạch dưới “\_” trong tên mã độc không bao giờ sử dụng dấu đơn, kép, khoảng trống.
  + - 1. File đặc biệt
         1. HTML

ClamAV có chứa một mã HTML đặc biệt binhg thường, giúp phát hiện lỗ hổng HTML. Chạy sigtool –html chuẩn hóa trên một tập tin HTML sẽ tạo ra các tập tin dưới đây:

Nocmomment.html: tập tin bình thường, với trường hợp thấp hơn tất cả các nhận xét và khoảng trắng bị xóa

Notags.html: giống như trên nhưng các thẻ HTML bị xóa

Cần tạo ra một chữ ký đối với các tập tin được tạo ra. Để loại bỏ khả năng

cảnh báo giả các mục tiếu cần được đặt giá trị là 3.

* + - * 1. Tập tin văn bản

Cũng giống như HTML các tập tin văn bản mã ASCII được bình thường hóa tất cả các khoảng trống và kiểm soát các ký tự bị xóa trước khi quét

Câu lệnh : clamscan – leave-temp để có một tập tin bình thường và sau đó tạo ra chữ ký với đích có kiểu là 7

* + - * 1. Tập tin thực thi di động đã nén

Nếu tập tin được nén bởi UPX, FSG, Petite hoặc một trình đống gói PE file khác được hỗ trợ bởi libclamav, cần chạy clamscan với tùy chọn --debug --leave- temps. Ví dụ xuất ra từ trình nén file FSG:

*LibClamAV debug: UPX/FSG/MEW: empty section found - assuming compression LibClamAV debug: FSG: found old EP @119e0*

*LibClamAV debug: FSG: Unpacked and rebuilt executable saved in*

*/tmp/clamav-f592b20f9329ac1c91f0e12137bcce6c*

Tiếp theo tạo một chữ ký từ :

*/tmp/clamav-f592b20f9329ac1c91f0e12137bcce6c*