Reverse Engineering

bismillah, Asslamualaikum, semalat malam. Kali ini saya akan membawakan kulgram tentang pengenalan dan dasar reverse engineering

Mungkin teman2 disini udah nggak asing yg mendengar kata Reverse Engineering, udah ada yg beberapa tau mungkin ya, dan juga ada yg belum paham maksud dari revere engineering itu. Apa sih reverse engineering itu, tujuannya untuk apa ?

Reverse engineering adalah suatu kegiatan yg bertujuan untuk mencari, memahami, dan mempelajari terhadap suatu benda atau produk yg sudah dibuat atau diciptakan.

Reverse engineering bisa dilakukan pada benda apapun, seperti barang2 elektronik, mesin dll. reverse engineering bisa dikatakan proses membongkar terhadap sesuatu yg sudah ada. misalnya kita ingin memahami proses cara kerja sebuah mesin yg sudah ada, untuk memahaminya pertama kita bongkar mesin tersebut, memahami komponen2nya. proses ini di namakan proses reverse engineering

seiring perkembangan komputer software yg kompleks saat ini, reverse engineering meluas ke ranah software. tujuannya apa ? salah satunya untuk mempelajari cara kerja software yg sudah dibuat. ini dinamakan software reverse engineering.

Jika software closed source ingin kita pahami cara kerjanya, karena kita tidak punya source code program, jadi kita harus *membongkarnya*

Apa manfaatnya ? salah satu yg paling kalian tahu mungkin software crack, dan keygen yg bisa membypass aktivasi pada sebuah software yg berbayar (jangan ditiru ya). untuk membuat keygen misalnya, kita harus memahami, bagaimana cara kerja program tersebut menghasilkan serial keynya, bagaimana proses pengecekannya ?. setelah memahami cara kerjanya, kita bisa membuat keygen untuk software tsb
masih banyak lagi manfaatnya.
misalnya RE untuk analisa malware, seorang malware analist harus mampu mengetahui cara kerja si malware tsb dengan proses reverse engineering.
RE untuk pentest, exploit2 yg di buat untuk sistem2 operasi closed source seperti windows, mac, ios. dibuat dengan proses reverse engineering terlebih dahulu, karena kita tidak mempunyai source code asli dari os tsb

contoh lain adalah, RE untuk meniru software orang lain, dengan mempelajari software tsb, kita bisa membuat ulang software baru yg hampir mirip dengan software orang lain tsb.
RE untuk memodifikasi program, contohnya adalah modding

Apa yg harus di kuasai ?.

Yg paling pnting adalah pemrograman, karena buat apa kita ngebongkar program kalau kita pun ga bisa pemrograman.

Misal kita akan mereverse program java, kita pun harus bisa bahasa java. Untuk mereverse program .net, kita juga harus bisa .net. dan untuk bahasa2 lainnya pun begitu

Kita juga harus tau, bagaimana2 compiler tersebut menghasilkan program. misalnya :
program yg kita buat dengan java, compiler akan menghasilkan java byte code
program yg dibuat dengan c#, vb .net, compiler akan menghasilkan kode CIL assembly
program yg dibuat dengan c, compiler akan menghasilkannya dalam bentuk binary (yg bisa kita ubah ke assembly dengan disassembler)

selain juga bahasa2 diatas, bahasa yg harus ketahui oleh reverser adalah bahasa assembly

bahasa2 pemrograman yg kita buat akan di ubah ke bahasa mesin oleh compiler agar bisa di eksekusi oleh komputer.
pada saat mengcompile kode program
pertama compiler akan menerjamahkan bahasa2 tsb kedalam bahasa assembly
selanjutnya bahasa assembly diubah menjadi bahasa mesin oleh assembler
dan bahasa mesin tsb akan diubah menjadi executable oleh linker

contohnya bahasa C
kode C -> assembly -> bahasa mesin/biner-> executable

[In reply to dword]
ini adalah proses, compiling dari bahasa C kedalam bentuk executable

contoh, saya punya kode bahasa C yg sangat sederhana

kita lakukan proses compile, disini saya menggunakan gcc.

setelah kode tsb dicompile, gcc akan menghasilkan file executable yg kita sudah coba jalankan sperti di gambar

inilah proses yg dilakukan gcc tsb

tapi kita juga bisa, memerintah gcc agar menghasilkan kode assembly saja (tidak menghasilkan program executable) dengan opsi -S

```
$ gcc -S hello.c
```

command diatas akan menghasilkan file hello.s yg berisi kode assembly

compiler menerjemahkan kode C ke dalam bahasa assembly

kalo yg belum pernah ngliat kode assembly mungkin agak kaget yg hehe. tapi ini lah kode assembly. kode assembly adalah bahasa tingkat rendah yg mendekati bahasa mesin. komputer hanya mengerti bahasa mesin. karena bahasa mesin terlalu tidak manusiawi untuk dipelajari, maka itu dibuat bahasa assembly, 1 intruksi bahasa assembly mewakili 1 intruksi bahasa mesin

karna ini lah yg akan kita hadapi, ketika kita mereverse program executable, kita akan dihadapi oleh kode2 assembly yg harus di pahami. jadi minimal kita harus bisa assembly

ingat ya, compiler akan menghasilkan program executable yg isinya bahasa2 mesin. jika kita membongkar program hello tsb dengah hexdump misalnya, makan akan sulit terbaca

Oh iya bahasa mesin adalah bahasa yg hanya di mengerti komputer, yg isinya hanya bilangan 0 dan 1. biasanya di tampilkan dalam bentuk hexa, saya biasa menyebeut intruksi2 bahasa mesin dengan sebutan opcode

[In reply to dword]
karna sulit terbaca, kita perlu tool yg bisa mengubah bahasa mesin ke dalam intruksi assembly. tool ini dinamakan disassembler

ini adalah contoh intruksi assembly dari program hello yg tadi kita buat.

digambar diatas saya menggunakan command
$ objdump -d hello
untuk melihat hasil disassembly.

objdump adalah salah satu tool yg bisa digunakan sebagai disassembler

ini adalah kode assembly dari fungsi main. sebenarnya output dari objdump sangat panjang, tapi saya me scrnshot bagian ini saja

lihat bagian tengah yg berisi bilangan2 hexa ?, itulah yg namanya bahasa mesin. sementara yg kanan adalah intruksi assemblynya. bahasa mesin ditampilkan bentuk hexa, karena jjika ditampilkan dalam biner akan sangan panjang dan sulit dibaca

contohnya intruksi "push rbp" dalam bahasa mesin akan menjadi 01010101 (55 dalam hexa)

selain disassembler, ada juga tool decompiler yg berfungsi mengubah kode assembly menjadi pseudo C
wait
hasil decompiling dengan tool snowman

liat diatas, kali ini akan lebih mudah terbaca, daripada disassembler tadi

dalam kode c kita tadi ada perintah printf("hello world"). tapi printfnya tidak terbaca

begitu juga dengan nama2 variable, tidak akan terbaca

untuk bahasa assembly mungkin tidak bisa di jelaskan dalam 1 malam ya, jadi bahasa assembly bisa kita bahas di lain waktu
untuk mengembangkan kemampuan reverse engineering, biasanya reverser akan mengerjakan challenge, challenge itu dibuat khusus untuk reverse engineering. challenge tsb biasanya kami menyebutnya "crackme", "reverseme" dll
Dan pahami juga assembly
Karena kalau gak paham assembly ntar challangenya ga bisa/paham utk dikerjain 😁

contoh challengenya misalnya, kita di beri sebuah program yg di suruh untuk menginputkan password. untuk mendapatkan password yg benar, kita harus membongkar program tsb dahulu, pahami programnya, dan kita harus mendapatkan password di dalam program tsb

iya, harus bisa assembly, walaupun ada decompiler yg langsung bisa mengubah assembly menjadi pseudo C, tetap harus mempelajari asm, walaupun saya juga sering melakukan decompile agar proses lebih cepat. tapi beberapa program dibuat dengan sengaja agar tidak bisa di decompile, jadi saya harus membaca kode assemblynya

selain disassembler dan decompiler, ada juga yg namanya debugger. debugger adalah tool yg dibuat untuk, mengetahui kondisi program pada saat dijalankan, untuk memahami perilaku si program tsb ketika dijalankan
tool debugger yg bisa kita gunakan adalah gdb (gnu debugger) yg sudah tersedia di kali linux
Debugger di win contohnya windbg
Ilmu reversing tidak bisa dijelaskan dalam semalam. tadi saya hanya menjelaskan dan mengenalkan sebagian dasar dari ilmu reversing saya, bahasa C, assembly dan tool2 yg bisa digunakan dalam reversing seperti disassembler, decompiler, debugger. saya akan membuka jika ada yg ingin bertanya. karena sudah 2 jam kita kulgram. selebihnya saya serahkan kepada moderator

dulu di tahun 2010 -an sempat ada versi GNU/Linux nya, tapi sekarang sudah ndak ada,

kira2 kalau saya belajar reversing supaya tujuan perangkat lunak tersebut bisa running secara native di GNU/Linux bisa mulai dari mana ya Pak? matur nuwun sakderenge =)

biar mudah, coba lakukan decompiling, pake ida pro. setelah dipahami, mungkin mas bisa remake versi linuxnya. btw ini program apa ?
untuk yg ingin mendalami reverse engineering, ada komunitas khusus yg mebahas tsb di @reversingid

https://id.wikipedia.org/wiki/IMindMap ; nah ini ketemu,

1.Apakah dari hasil dari disassembly yang jadi source assembly, bisa di compile lagi
Dan berfungsi sama seperti program aslinya?

2.
dword:
contohnya intruksi "push rbp" dalam bahasa mesin akan menjadi 01010101 (55 dalam hexa)

Ini selalu dengan kode hexa 55?

3.dword:
selain disassembler, ada juga tool decompiler yg berfungsi mengubah kode assembly menjadi pseudo C

Pseudo c itu sintaknya beda banget sama c++ ya?

https://www.ilmuhacking.com/programming/belajar-assembly-di-linux

iya mas "push rbp" itu akan selalu 0x55,

pseudo c itu kode c juga, tapi dia agak terlihat beda, karena itu hasil dari kode assembly menjadi kode c. decompiler ga bisa spenuhnya bisa nerjemahin assembly ke c. jadi itu blum spenuhnya bahasa c lengkap, dinamakan pseudo c

1.Apakah dari hasil dari disassembly yang jadi source assembly, bisa di compile lagi
Dan berfungsi sama seperti program aslinya?

untuk yang ini bang?

misalnya tidak kita rubah code assembly nya

belum tentu bisa mas, karena kode assembly yg dihasilkan compiler akan banyak sesuatu yg ditambahkan, seperti loader, shared library, dll. kalo kita cuman ambil kode assemblynya dan coba mengcompilenya lagi mungkin akan error
smoga bermanfaat ya kulgram yg saya sampaikan, walaupun kita ga banyak waktu >_< hihihi dan kalo ada yg kurang, dan cara menjelaskan saya juga kurang dimengerti mohon di maafkan 😄

https://www.tutorialspoint.com/assembly_programming/
