r2 (radare2) - guía de supervivencia

INCIDE - Ref.r2-gs-20200412

12 de abril de 2020

Capítulo 1

historial de versiones

Fechar	Versión	Autor/Editor	Razón
12 abril 2020	1.0	Abraham Pasamar	Creación documento

Capítulo 2

r2 (radare2) - guía de supervivencia

2.1. ayuda r2

- radare2 site -> https://rada.re/n/
- github -> https://github.com/radareorg/radare2
- r2 book -> https://radare.gitbooks.io/radare2book/content/
- r2 wiki -> https://r2wiki.readthedocs.io/en/latest/
- r2con videos -> https://www.youtube.com/results?search_query=r2con
- alias muy recomendable: `alias r2help='r2 -q -c '\''?*~...'\'' -'` (ponlo en tu `~/.bashrc` file)
- ver ayuda de r2: `$ r2 -h`

2.2. invocando radare2

- Ejecutar r2 con un `malloc 512`

```
$ r2 -
```

- Ejecutar r2 sin abrir ningún fichero

```
$ r2 --
```

- Ejecutar r2 usando los datos de `stdin`

```
$ echo "hola"| r2 =
```

- Ejecutar r2 abriendo un archivo (`$ r2 <file>`)

```
$ r2 /bin/ls
```

2.2.1. el prompt de r2

`[0x00000000]>` <- offset en el que se encuentra r2 en este momento, al cargar un binario el offset será el punto de entrada del binario (diferente para cada binario). Si no se usa el falg -n al invocar r2 el offset se situará en `[0x00000000]>`

2.3. hashing

- Cálculo del hash MD5

`[]> ph md5` <- calcula el hash md5 de un bloque, no del binario

- Cálculo del hash MD5 del binario ls

`$ r2 -n /bin/ls` <- cargarlo en modo raw, el offset será 0x0
`[0x00000000]> ph md5 $s` <- se calcula el hash MD5 desde el offset 0x0 hasta el final ($s=file size)

2.4. entropía

- Cálculo de la entropía del archivo

`[0x00000000] ph entropy $s` <- desde offset 0x0 representación histograma entropía

```
$ r2 /bin/ls
[]> p==e
```

2.5. modo quiet

- Con `-q` se invoca el modo quiet mode (no prompt)

```
$ r2 -q
```

- Ejecutar un comando (entropía) sin entrar en r2 (sin prompt)

```
$ r2 -qnc 'ph entropy $s' /bin/ls
```

2.6. analisis de un. archivo PE

- Abrir archivo con radare2

```
$ r2 hello_world
```

- Ver información

```
[]> i
```

- Ver información reducida

```
[]> iq
```

- Ver secciones

```
[]> iS
```

- Ver imports

```
[]> ii
```

- Ver strings (en secciones de datos)

```
[]> iz
```

- Ver strings (en todo el binario)

```
[]> izz
```

- Ir a una dirección o “simbolo” (s: seek)

```
[]> s main
```

- Analizar binario

```
[]> aaa
```

- Desensamblar 5 instrucciones

```
[]> pd 5
```

- Desensamblar función (necesirio analisis previamente `aaa`)

```
[]> pdf
```

- Ver referencias cruzadas

```
[]> ax
```

- Filtrar referencias cruzadas (con grep o ~)

```
[]> ax|grep main
[]> ax~main
[]> ax~main+~CALL
```

- Ver callgraph (función actual)

```
[]> agc
```

- Ver callgraph (binario completo)

```
[]> agC
```

- Ver callgraph (y generar un archivo imagen)

```
[]> agcw
[]> agCw
```

2.7. modo visual

- Modo visual (usar p para cambiar vista)

```
[]> V
```

- Modo visual bloques flujo (usar p para cambiar vista)

```
[]> VV
```

- Paneles

```
[]> v
```

2.8. guardar información análisis r2

2.8.1. projects

- Listar proyectos

```
P
```

- Guardar proyecto

```
Ps
```

- Abrir proyecto

```
Po
```

2.8.2. usar fichero de comandos

```
$ r2 -i commands_file.txt /bin/ls
```

- Historial de comandos (copiar comandos del historial y crear fichero para utiliziar con `r2 -i`)

```
cat $home/.cache/radare2/history
```

2.9. variables

2.9.1. comando 'e'

- El comando `e` sirve para configurar las variables de r2. Escribe `e??` para ver la lista de variables y su descripción

```
[0x004014a0]> e??
anal.a2f: Use the new WIP analysis algorithm (core/p/a2f),
   anal.depth ignored atm
anal.arch: Select the architecture to use
.....
.....
.....
zoom.maxsz: Zoom max size of block
zoom.to: Zoom end address
```

- Es un comando con una salida muy larga. Utiliza `grep` o `~`

```
[]> e??~fortunes cfg.fortunes: If enabled show tips at
start cfg.fortunes.clippy: Use ?E instead of ?e cfg.fortunes
.tts: Speak out the fortune cfg.fortunes.type: Type of
fortunes to show (tips, fun, nsfw, creepy) ## fortunes
```

- Las fortunes se controlan a través de las variables de configuración de radare

```
[]> e cfg.fortunes [TAB for options]
cfg.fortunes          cfg.fortunes.clippy   cfg.fortunes.
   tts      cfg.fortunes.type
```

- Quitar las fortunes

```
[]> e cfg.fortunes=False
```

- Poner las fortunes

```
[]> e cfg.fortunes=False
```

2.10. guardar las preferencias de r2

- Utiliza ~/.radare2rc para guardar tus preferencias de r2 de forma permanente

```
$ cat ~/.radare2rc
eco onedark
e scr.color=1
e scr.utf8 = true
e cfg.fortunes=false
```

2.11. Temas (Themes)

- Utiliza `eco` para listar los temas disponibles.

```
[]> eco
  default
  xvilka
  lima
  bright
> onedark
  consonance
  rasta
  basic
  solarized
  ogray
  tango
  ...
```

- Selecciona un tema

```
[]> eco lima
```

2.12. r2pipe

- Instalar r2pipe

```
$ pip install r2pipe
```

- Ejemplo r2pipe (python) para analizar binario y listar funciones

```
import r2pipe
r2 = r2pipe.open("/bin/ls")
r2.cmd('aa')
print(r2.cmd("afl"))
print(r2.cmdj("aflj"))  # evaluates JSONs and returns an
    object
```

2.13. plugin r2 de ghidra

2.13.1. instalación

```
$ r2pm init
$ r2pm update
$ r2pm install r2ghidra-dec
```

2.13.2. decompilando con ghidra

```
$ r2 /bin/ls
$ aaa
$ pdg
```

2.13.3. mensajes de error

- Evitando el error `version plugin mismatch messages`

```
$ r2pm -i `r2pm -l`
```