Lehrstuhl für Systemsicherheit
Prof. Dr. Thorsten Holz

Vorlesung Betriebssystemsicherheit
Wintersemester 2016/2017
Ruhr-Universität Bochum

3. Übung

28. November 2016
Abgabe der Hausaufgaben: per Moodle bis zum 12. Dezember 2016, 16 Uhr
Bei Fragen und Problemen können Sie sich per E-Mail/Moodle an die Betreuer wenden.

In diesem Blatt sollen Sie eine Reihe von Schwachstellen in verwundbaren Programmen ausnutzen, um Kontrolle über ein Programm zu erlangen. Benutzen Sie dazu die speziell präparierte 32bit Linux-VM (Link siehe Moodle). Auf anderen Linux-Systemen wird die Entwicklung eines Exploits höchstwahrscheinlich durch Randomisierung und Non-Executable-Stack deutlich erschwert. Für alle Aufgaben haben wir bereits kompilierte Binärdateien erstellt, für die der jeweilige Exploit erstellt werden soll. Die Dateien für Aufgabe 1-3 befinden sich in einer separaten ZIP-Datei. Das Passwort des Standardbenutzers in der VM lautet `hackmeifyoucan`.

Aufgabe 1 (authenticate.c) (2 Punkte)

Die Quelldatei `authenticate.c` enthält eine einfache Authentifizierungsroutine. Aufgrund eines Programmierfehlers ist es möglich, sich trotz eines vermeintlich falschen Passworts zu authentifizieren. Geben Sie einen solches Passwort an und erklären Sie kurz, wo der Fehler im Quellcode liegt. Erläutern Sie außerdem, was passiert, wenn die Zeilen 6 und 7 vertauscht werden. Funktioniert der Angriff dann immer noch? Erläutern Sie auch, wie man diese Schwachstelle verhindern kann.

Aufgabe 2 (Basic Overflow) (3 Punkte)

Nutzen Sie den Stack Overflow in der Datei `basic_overflow.c` aus, um durch Übergabe eines entsprechenden Kommandozeilenparameters eine Shell zu erzeugen. Den Shellcode können Sie dabei beliebig wählen. Shellcodegeneratoren befinden sich z.B. im Metasploit Framework. Bitte achten Sie zusätzlich darauf, dass der Exploit stabil läuft, d.h. auch nach dem Reboot des Systems funktioniert.

Schreiben Sie den Exploit als kleines Programm, das `basic_overlow` aufruft und laden Sie den Quelltext bei Moodle hoch. Beschreiben Sie außerdem kurz als *Kommentar im Quelltext*, wie Ihr Exploit funktioniert, d.h. an welcher Stelle im Shellcode sich die Rücksprungadresse befindet und wie genau die Shell erzeugt wird. Um Probleme beim Korrigieren durch verschobene Adressen vorzubeugen, gehen wir davon aus, dass sich die Datei `basic_overlow` im Ordner `/home/user/aufgabe2` befindet und auch der Exploit aus diesem Ordner heraus ausgerufen wird.

Aufgabe 3 (Return-to-Libc) (3 Punkte)

Nutzen Sie die Schwachstelle in der Datei `basic_overflow_fread.c` aus, um eine Return-To-Libc Attacke auszuführen. Dabei soll die Standard-C Funktion `system` in `libc` mit dem Parameter `"touch owned"` angesprungen werden. Wenn die Attacke erfolgreich ausgeführt wurde, sollte also anschließend die Datei `owned` erzeugt worden sein.

Schreiben Sie den Exploit wieder als eigenes Programm, das `basic_overflow_fread` aufruft und laden Sie den Quellcode in Moodle hoch. Beschreiben Sie außerdem kurz als Kommentar im Quelltext, welche Adressen für den Angriff benötigt werden und wo sich diese in Ihrem Shellcode befinden. Um Probleme beim Korrigieren durch verschobene Adressen vorzubeugen, gehen wir davon aus, dass sich die Datei `basic_overflow_fread` im Ordner `/home/user/aufgabe3` befindet und auch der Exploit aus diesem Ordner heraus ausgerufen wird.

Hinweis: Die Basisadressen von geladenen Bibliotheken kann man durch Ausgabe von `/proc/pid/maps` oder mit Hilfe von `gdb` (info shared) erhalten. Die Basisadresse von `libc` ist für verschiedene Programme *nicht* identisch.

Aufgabe 4 (BMP Validator) (Bonusaufgabe, 0 Punkte)

Im Verzeichnis `vuln1` in der VM befindet sich der *BMP Validator*. Das Programm überprüft die Checksumme von BMP Dateien, das Format ist dabei rein fiktiv und ergibt sich aus dem Quelltext. Erzeugen Sie eine BMP-Datei, die beim Öffnen mit dem BMP Validator einen Buffer Overflow auslöst und eine Shell erzeugt.

Laden Sie diese Datei bei Moodle hoch und beschreiben Sie wieder als Kommentar (in einer separaten Textdatei oder als Kommentar im Quelltext), wie sich die Schwachstelle ausnutzen lässt, d.h. wie die vorherigen if-Abfragen umgangen werden können (Stichwort: Vorzeichen).