Crashing vulnerability in Mozilla Firefox

Table of Contents

Introduction

Mozilla Firefox is a widely used web browser. Latest version of Mozilla Firefox 32.0

, is effected by buffer over flow vulnerability.

What is Buffer Over flow?

Buffer overflow has been called the “attack of the decade”, and it is widely agreed that buffer overflow attacks remain the most commonly exploited vulnerability in computing systems today . A buffer overflow generally occurs due to a programming flaw that allows more data to be written to a buffer than the buffer was designed to hold. As a result, the memory following the buffer is overwritten. This overflow can overwrite useful information, but what makes these attacks particularly damaging is that an attacker can often overflow the buffer in such a way that code of the attacker’s choosing executes on the victim’s machine. Such buffer overflow attacks are somewhat delicate, and developing such an attack is usually time-consuming, and requires a significant amount of trial and error. In this project, we provide a method that takes advantage of these factors in order to reduce the likelihood of a widespread buffer overflow attack.

In computer science, a buffer is usually a contiguous computer memory block to store data of the same type. This data can be integers, floating points, characters, or even user defined data types. In most computer languages, a buffer is represented as an array.

Impact of Attack

Severe

Common Consequences

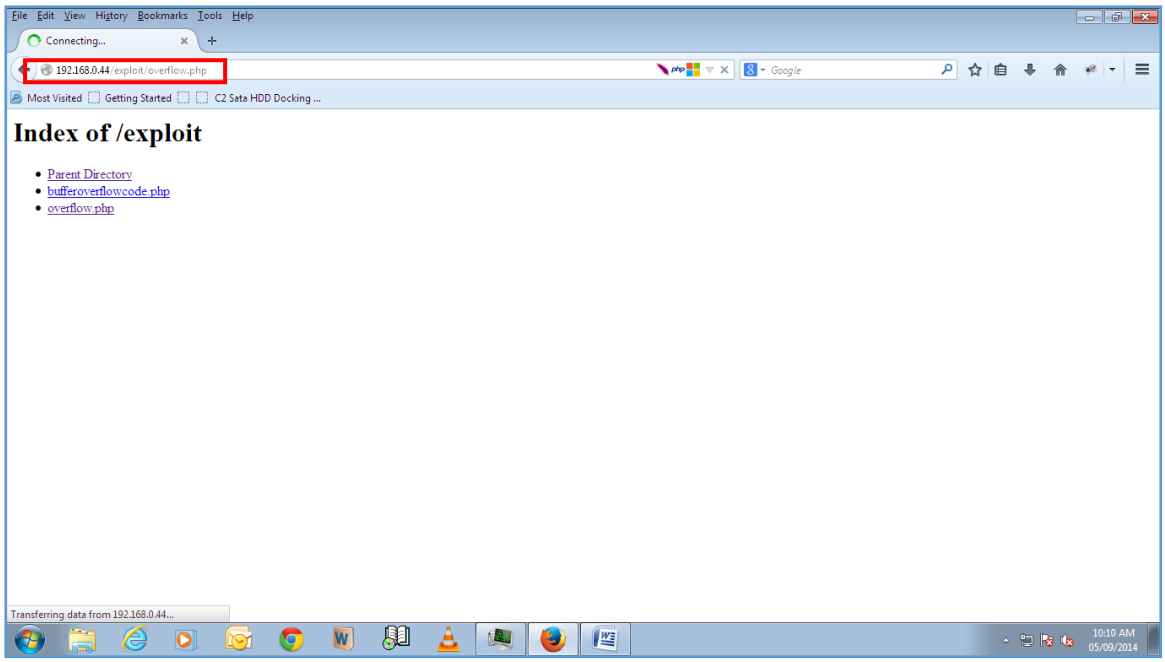
Scope	Effect
Availability	**Technical Impact:** *DoS: crash / exit / restart; DoS: resource consumption (CPU); DoS: resource consumption (memory)* Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop.
Integrity **Confidentiality** **Availability** **Access Control**	**Technical Impact:** *Execute unauthorized code or commands; Bypass protection mechanism* Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program's implicit security policy.

Demonstrative Example:

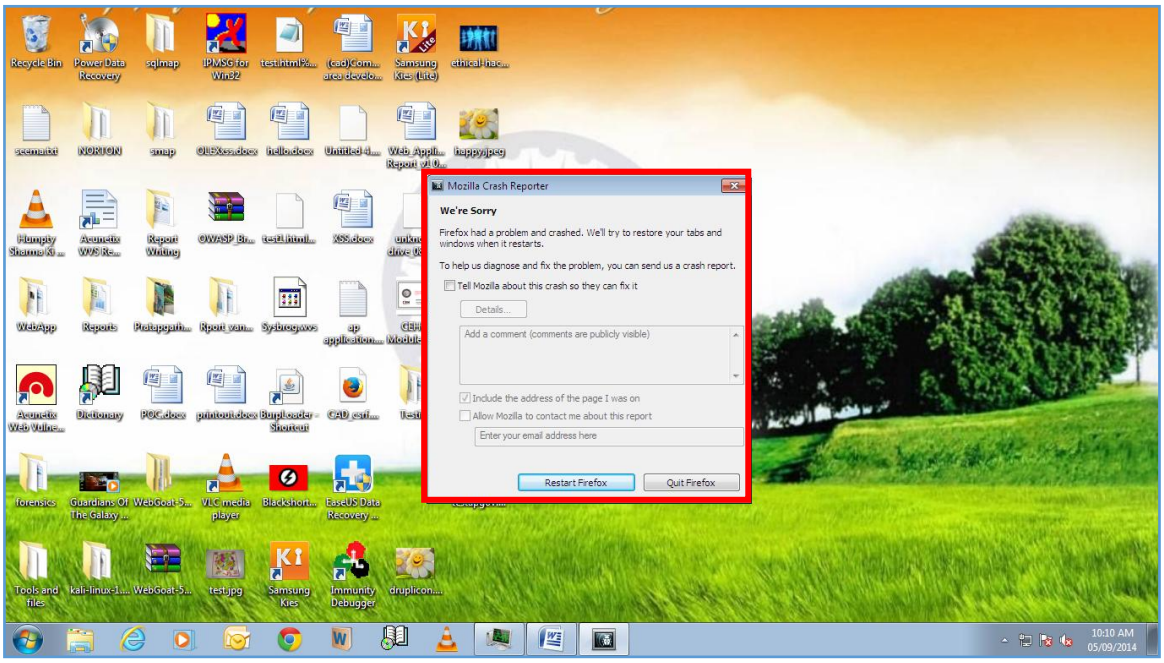
1. Access the file **overflowcode.php** that is locally hosted on **another system** shown in the snapshot:

    - File contains a php code of **infinite loop code** of file is given below**:**

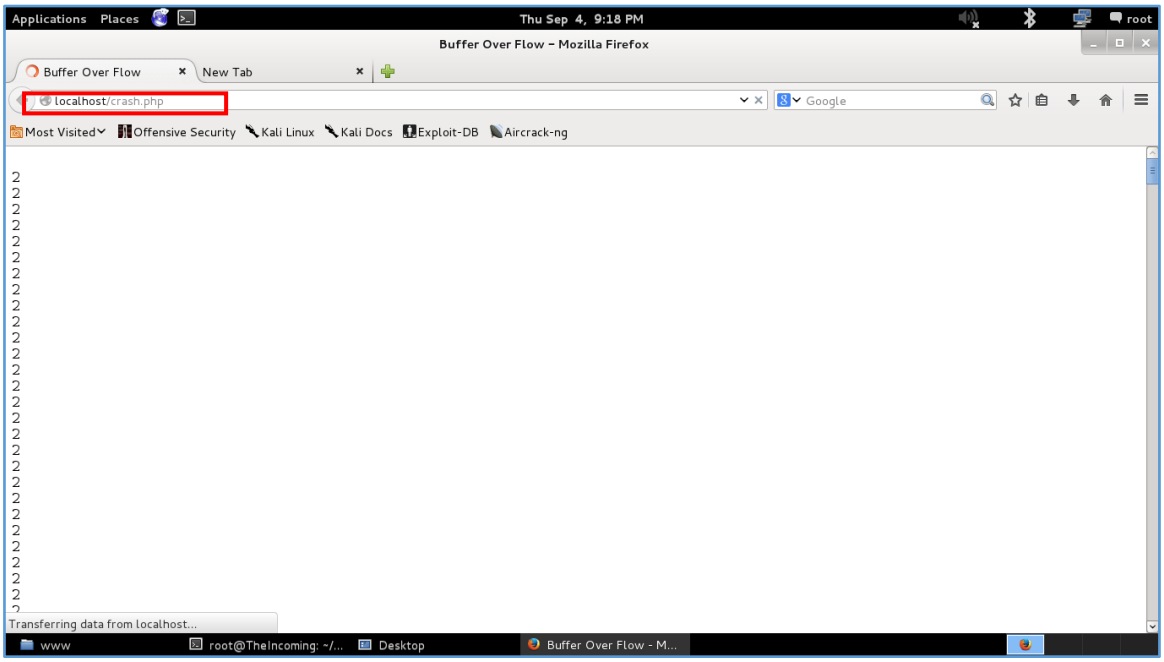
```
<html><head><title>Buffer Over Flow </title></head><body>
<?php
$c=1;
$a=0;
while($c<=10){
echo $a;
echo "<br>";
$a++;
}
?>
</body>
</html>
```



2. Mozilla firefox get crashed as shown in the snapshot:



Note : It is crashing regardless of operating system. Snapshots of kali Linux machines are given below:



Note: Here name of the php file is crash.php which is locally hosted on same machine.

Solution

Various techniques have been used to detect or prevent buffer overflows, with various tradeoffs. The most reliable way to avoid or prevent buffer overflows is to use automatic protection at the language level. This sort of protection, however, cannot be applied to legacy code, and often technical, business, or cultural constraints call for a vulnerable language.

Remark

Please take this vulnerability seriously.

About me

Nidhi Singh Cyber Security Professional. My area of interest includes Network & Web Penetration ,Testing, attack research. My responsibilities include ‘Discovering Vulnerability’, ‘Exploitation’, and 'Reporting’.