# Shellcode Injection Analysis

# Shellcode Injection into a Remote Process

## Overview

This program demonstrates a form of process injection where custom shellcode is injected into a remote process (`mspaint.exe`) and executed using `CreateRemoteThread`.

## 1. Shellcode Definition

The `myShellCode` variable holds 520 bytes of x64 shellcode. This shellcode is presumably designed to show a message box or perform some payload, although its exact behavior depends on the encoded machine instructions.

## 2. Function: SearchForProcess

- **Purpose**: Locates the PID (Process ID) of a running process with the specified name.

- **How it works**:

  - Creates a snapshot of all processes using `CreateToolhelp32Snapshot`.

  - Iterates through each process entry using `Process32First` and `Process32Next`.

   - Compares each process's executable name to the target using `lstrcmpiA` (case-insensitive string comparison).

  - Returns the PID of the matching process.

Shellcode Injection Analysis

## 3. Function: ShellInject

- **Purpose**: Injects and executes shellcode in a remote process.

- **Steps**:

  - Allocates memory in the target process (`VirtualAllocEx`) with `PAGE_EXECUTE_READ` permission.

  - Writes the shellcode into the allocated memory using `WriteProcessMemory`.

  - Creates a remote thread in the target process starting at the shellcode address (`CreateRemoteThread`).

  - Waits up to 500ms for the thread to finish and closes the handle.

## 4. Main Function

- **Steps**:

  - Calls `SearchForProcess` to find the PID of "mspaint.exe".

  - If found, opens the process with necessary permissions (`OpenProcess`).

  - Calls `ShellInject` to write and execute shellcode in the target process.

  - Closes the process handle afterwards.

## Security Implications

This technique is commonly used in malware for code injection and should be carefully monitored by endpoint detection systems. Defensive measures include:

- Blocking `VirtualAllocEx` + `WriteProcessMemory` + `CreateRemoteThread` patterns.

Shellcode Injection Analysis

- Monitoring execution of unsigned or suspicious shellcode.