Dangling Pointer

Jonathan Afek, 1/8/07, BlackHat USA
Table of Contents

- What is a Dangling Pointer?
- Code Injection
- Object Overwriting
- Demonstration
- Remediation
- Summary
- Q&A
What is a Dangling Pointer?

Invalid Pointer:

Internet Information Services has encountered a problem and needs to close. We are sorry for the inconvenience.

If you were in the middle of something, the information you were working on might be lost.

Please tell Microsoft about this problem.
We have created an error report that you can send to help us improve Internet Information Services. We will treat this report as confidential and anonymous.

To see what data this error report contains, click here.

Object  Object  Object

Application Code:

Pointer3 = new Object();
...
delete Pointer3;
...
Pointer3->func();

Overwrite the object
What is a Dangling Pointer?

- **Assembly**
  - Memory Layout
    
    ![Assembly code]
    
    - Registers
    - Assembly code
Where are We

- What is a Dangling Pointer?
- **Code Injection**
- Object Overwriting
- Demonstration
- Remediation
- Summary
- Q&A
Code Injection – The Layout of an Object

- **Class_A:**
  ```cpp
  class Class_A {
  public:
  virtual long vfunc_A1();
  virtual long vfunc_A2();
  virtual long vfunc_A3();
  static void sfunc_A();
  void funcA();
  }
  ```

  ```assembly
  ... this.vfunc_A3(); ...
  MOV EAX, [ECX]
  CALL [EAX + 8]
  ... 
  ```
Exploit Overview:
- Free the Object
- Overwrite the Object
- Execute a Virtual Function
Code Injection – The Double Reference Exploit

- **Object Allocated**
- **Object De-allocated**
- **Overwriting Object**
- **VFunc3 Executed**

Application Code:

```
a = new A();
MOV EAX, [ECX]
CALL [EAX + 8]
...```

CALL/JMP ECX
Where are We

- What is a Dangling Pointer?
- Code Injection
- **Object Overwriting**
- Demonstration
- Remediation
- Summary
- Q&A
Object Overriding

- Allocation Implementation
  - C-Runtime heap
  - C-Runtime functions
    - Malloc
    - Free
    - New
    - Delete
    - Etc.
Object Overriding

- Allocation implementation details
  - Lookaside List: Cache De-allocated Memory
    - A list for each size (8-1024) (8)
    - First Allocation Priority
Object Overriding

- Exploit Review
- Overwriting
  - Search for Allocations
    - Disassembly
    - Same Size
    - Controllable Content

Application Code:

```cpp
Object Object3 = new Object();
...
delete Pointer3;
...
Pointer3->func();
```

Overwrite the object
Object Overriding – The VTABLE Exploit

- Empty the Lookaside List
- Allocate a Buffer
- Insert Content
- Free the Buffer
- Free the Object
- Execute a VFunc

```
MOV EAX, [ECX]
```

**Diagram:**
- ECX – Original Object
- EAX – “VTABLE”
- “VTABLE” Pointer
- Original Object
- SHELLCODE
- Pointer
- Rest of SHELLCODE
- CALL/JMP EAX
Object Overriding – The Lookaside Exploit

- Empty the Lookaside
- Allocate Two Buffers
- Insert Shellcode
- Free One Buffer
- Free the Other
- Free the Object
- Trigger the Bug

GAME OVER!!!
Object Overriding – The Lookaside Exploit

- Executing NULL – NO Problem

<table>
<thead>
<tr>
<th>Address</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>ADD BYTE PTR DS:[EAX],AL</td>
</tr>
<tr>
<td>0000</td>
<td>ADD BYTE PTR DS:[EAX],AL</td>
</tr>
<tr>
<td>0000</td>
<td>ADD BYTE PTR DS:[EAX],AL</td>
</tr>
<tr>
<td>0E34</td>
<td>SYSFNTFR</td>
</tr>
</tbody>
</table>
Summary

- **Double Reference Exploit**
  - Controllable First DWORD
  - Static Address

- **VFTABLE Exploit**
  - Controllable Allocations
  - No First DWORD
  - Static Address

- **Lookaside Exploit**
  - Controllable Allocations
  - No First DWORD
  - No Static Address
  - Destructor Execution
Where are We

- What is a Dangling Pointer?
- Code Injection
- Object Overwriting
- **Demonstration**
- Remediation
- Summary
- Q&A
Demonstration

- Putting it Together
  - De-Allocate
  - Inject
  - Trigger
Where are We

- What is a Dangling Pointer
- Code Injection
- Object Overwriting
- Demonstration
- **Remediation**
- Summary
- Q&A
Remediation

- Known Protection Mechanisms
  - NX Bit
  - ASLR
- VFTABLE Sanitation
- Safe Programming
Summary

- Technical Background
  - Memory Allocations
  - Objects Implementation
- Exploits
  - Double Reference Exploit
  - VFTABLE Exploit
  - Lookaside Exploit
- Demonstration
  - Microsoft IIS 5.1
- Dangling Pointer
  - Only Object Oriented Objects
More Information

- www.Watchfire.com
Questions

- Ask Away...