



Dangling Pointer

Jonathan Afek, 2/8/07, BlackHat USA

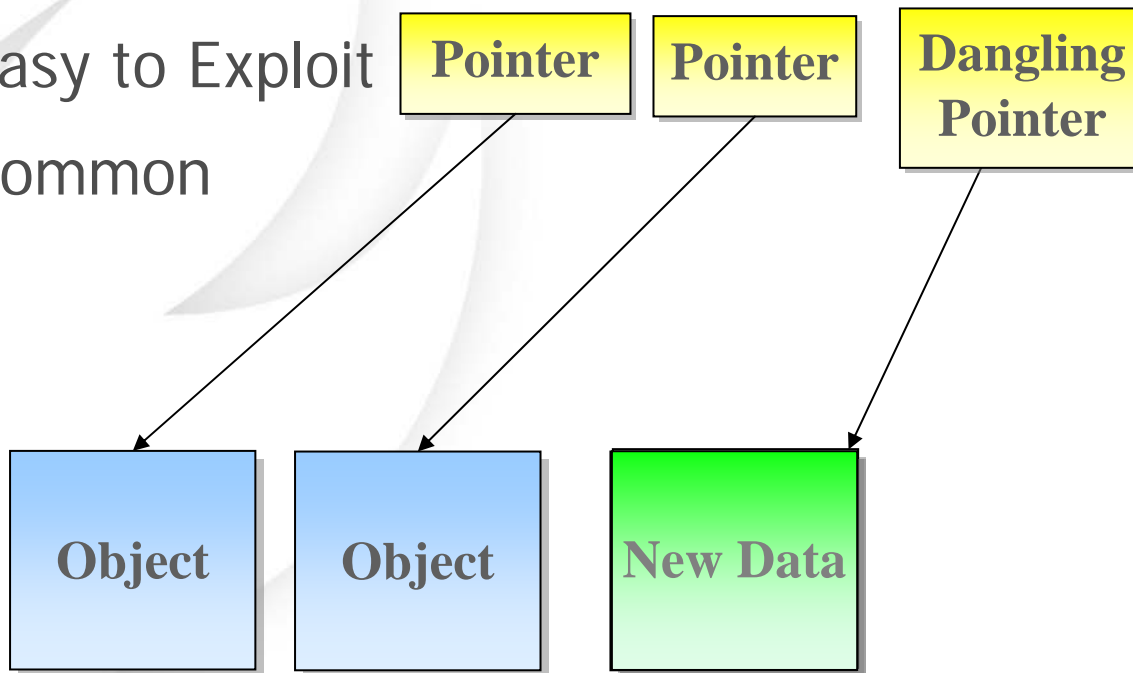
Table of Contents

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

What is a Dangling Pointer?

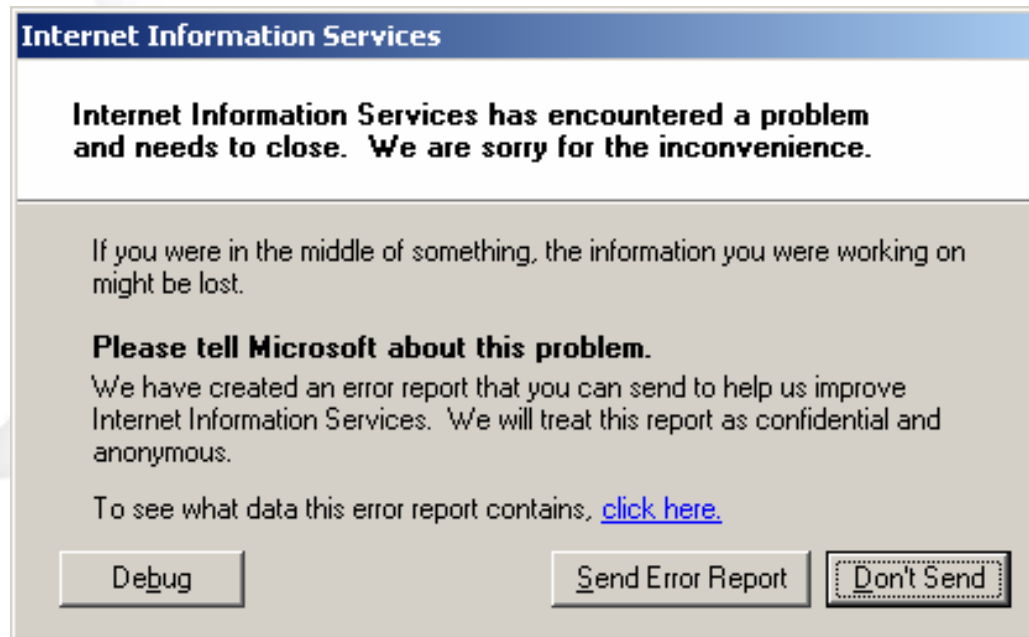
Invalid Pointer:

- Dangerous
- Easy to Exploit
- Common



What is a Dangling Pointer? – An Example

- Results:
Crash



What is a Dangling Pointer? – An Example

- Debugger View

5AA5919A	. 5D	POP EBP
5AA5919B	L. C2 0C00	RETN 0C
5AA5919E	CC	INT3
5AA5919F	CC	INT3
5AA591A0	CC	INT3
5AA591A1	CC	INT3
5AA591A2	CC	INT3
5AA591A3	§ 8BFF	MOV EDI,EDI
5AA591A5	. 56	PUSH ESI
5AA591A6	. 8BF1	MOV ESI,ECX
5AA591A8	. 8B4E 20	MOV ECX,DWORD PTR DS:[ESI+20]
5AA591AB	. 57	PUSH EDI
5AA591AC	. 6A 01	PUSH 1
5AA591AE	. C706 F38AA25A	MOV DWORD PTR DS:[ESI],w3svc.??_7WAM_RE
5AA591B4	. 8B01	MOV EAX,DWORD PTR DS:[ECX]
5AA591B6	. 56	PUSH ESI
5AA591B7	. FF50 0C	CALL DWORD PTR DS:[EAX+C]
5AA591BA	. 8B4E 20	MOV ECX,DWORD PTR DS:[ESI+20]
5AA591BD	. E8 E496FFFF	CALL w3svc.?Dereference@CWamInfo@@QAEXX
5AA591C2	. 8BCE	MOV ECX,ESI
5AA591C4	. E8 4BDAFFFF	CALL w3svc.?DoCleanupOnDestroy@WAM_REQU
5AA591C9	. 85C0	TEST EAX,EAX

01004340	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
01004350	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
01004360	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00

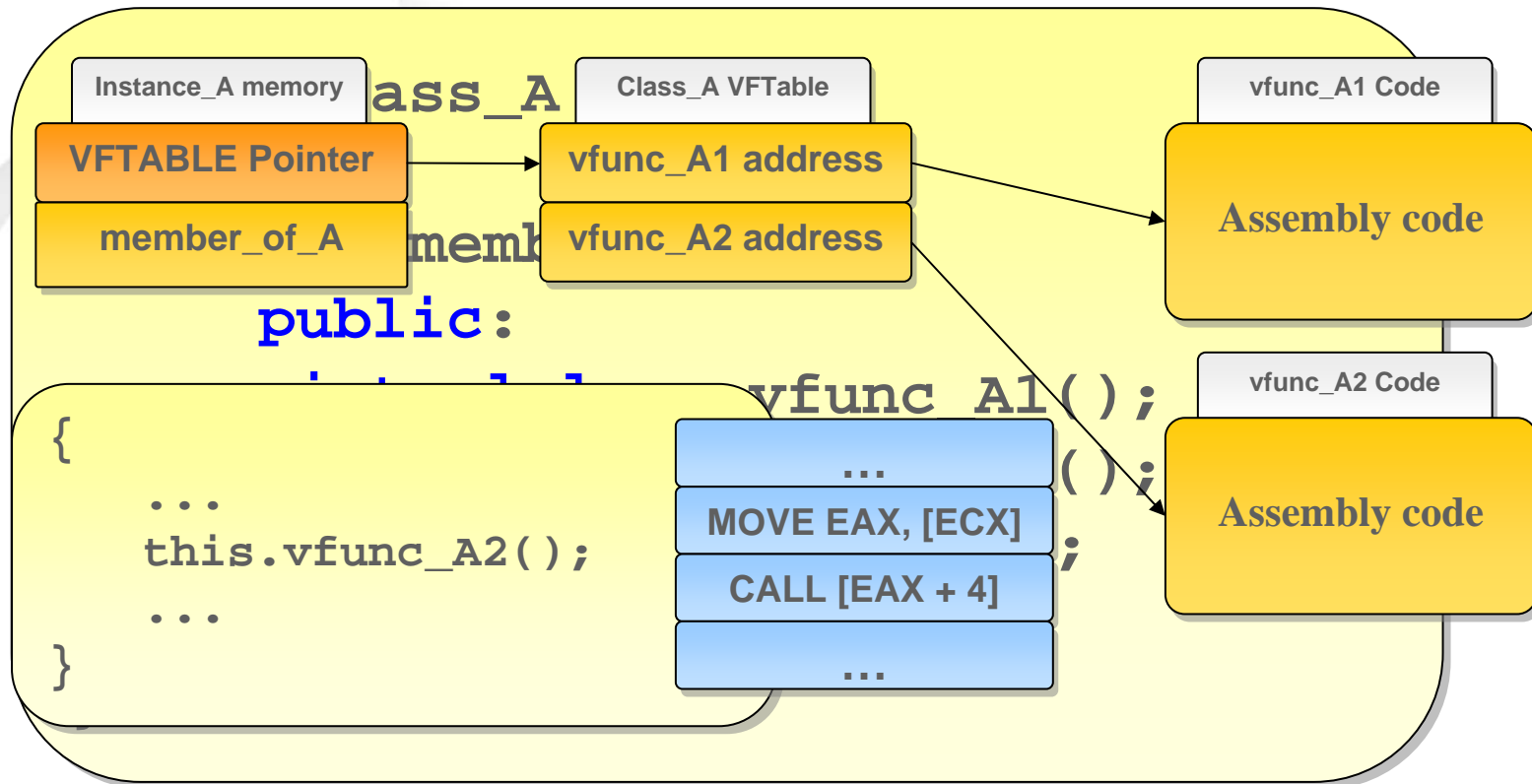
Access violation when reading [0000000C] - use Shift+F7/F8/F9 to pass exception to program

Where are We

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

Code Injection – The Layout of an Object

- Class_A:



Code Injection – The Double Reference Exploit

Exploit Overview:

- Free the Object
- Override the Object – covered later
- Execute a Virtual Function

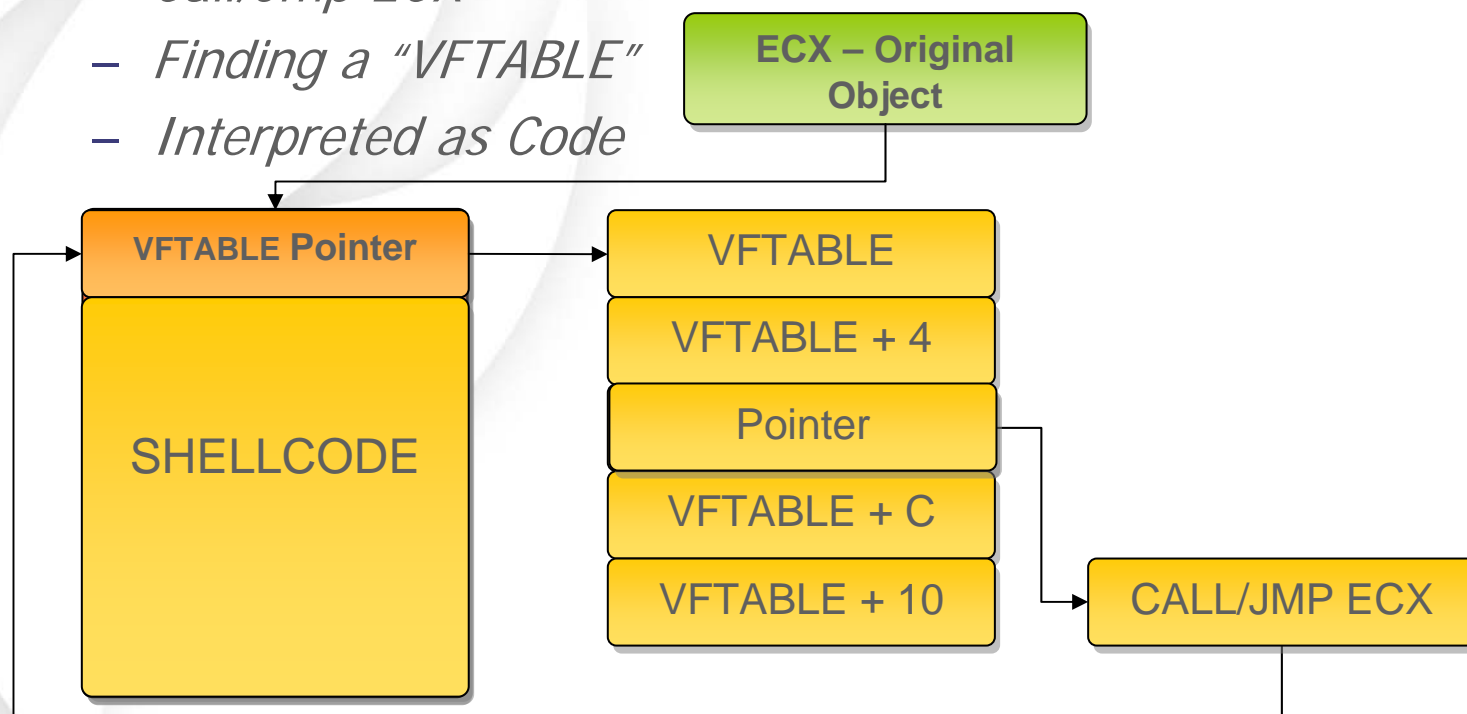
Code Injection – The Double Reference Exploit

- *Injecting Code*

- *Free the Object*
- *Shellcode*
- *Call/Jmp ECX*
- *Finding a "VFTABLE"*
- *Interpreted as Code*

- *Continue*

- *Automation*



Code Injection – Double Inheritance

- Multiple Inheritance

```
class Inherited: public Class_A, public Class_B
{
public:
    virtual int vfunc_A2();
    virtual int vfunc_B2();
};
```

Class_A::vfunc_A1

Assembly code

Inherited::vfunc_B2

Assembly code

Where are We

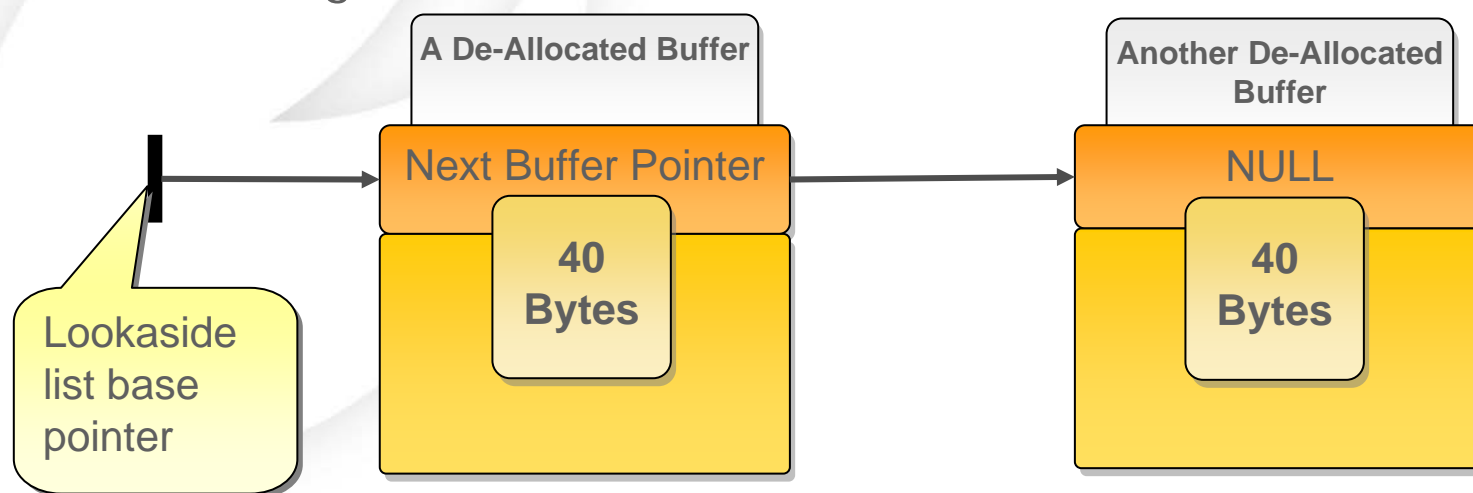
- What is a Dangling Pointer?
- Code Injection
- **Object Overriding**
- Demonstrations
- Remediation
- Summary
- Q&A

Object Overriding

- Allocation Implementation
 - Numerous heaps
 - Two Default heaps
 - Different API
 - C-Runtime functions
 - Malloc
 - Free
 - New
 - Delete
 - Etc.

Object Overriding

- Allocation implementation details
 - Lookaside List
 - A list for each size (8-1024) (8) and for each heap
 - First Allocation Priority
 - Merges



Object Overriding

- And Finally – Overriding
 - Search for Allocations
 - Static Analysis
 - Method: Disassembly
 - Restriction: Static Size
 - Validation: Controllable Content
 - Usage: Causing the Allocation
 - Dynamic analysis
 - Method: API Breakpoints
 - Restriction: Static/Dynamic Size
 - Validation: Controllable Content

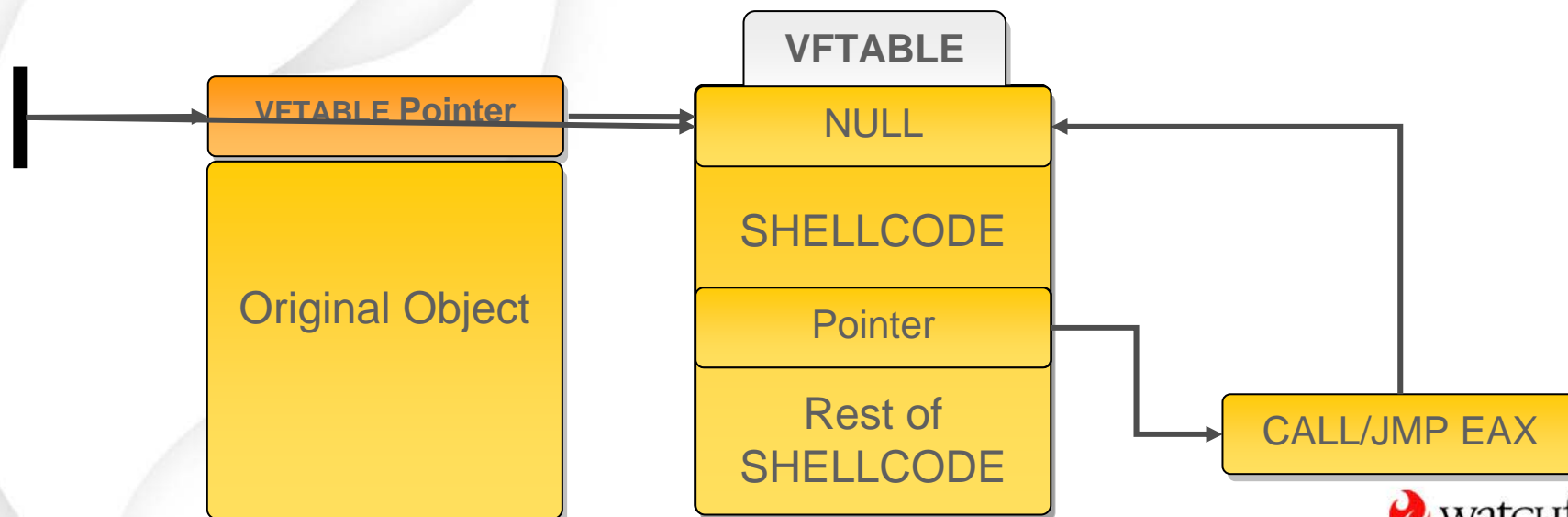
Object Overriding – The VFTABLE Exploit

- Exploitation:

- Empty the Lookaside List
- Allocate a Buffer
- Insert Content
- Free the Buffer

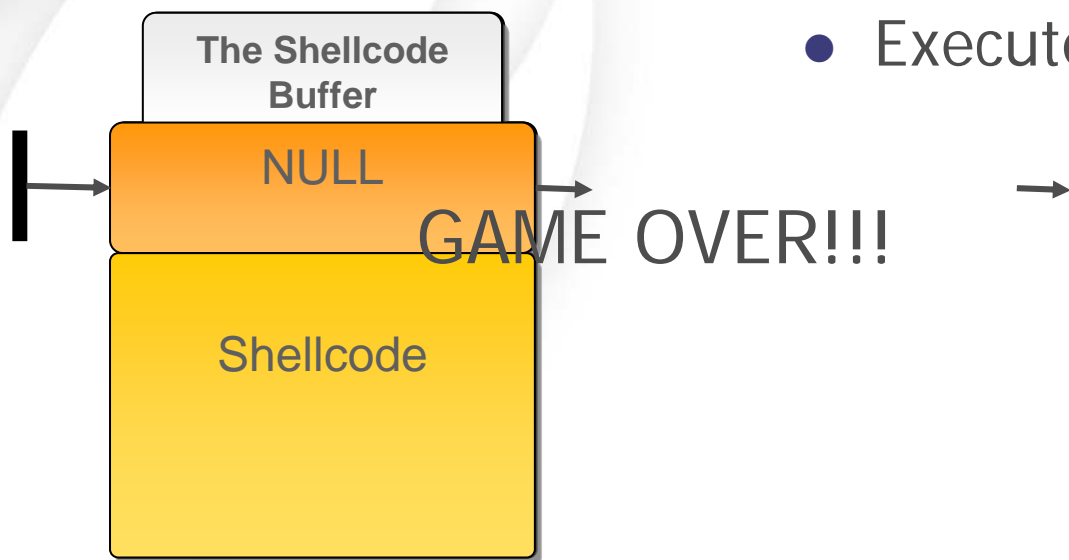
- Continue:

- Free the Object
- Execute a VFunc



Object Overriding – The Lookaside Exploit

- Empty the Lookaside
- Allocate Two Buffers
- Insert Shellcode
- Free One Buffer
- Free The Other
- Free The Object
- Execute the Destructor



Object Overriding – The Lookaside Exploit

- Executing NULL – NO Problem

```
70      NOP
0000    ADD BYTE PTR DS:[EAX],AL
0000    ADD BYTE PTR DS:[EAX],AL
0000    ADD BYTE PTR DS:[EAX],AL
0E34    SYSENTER
```

Summary

- Summary
 - Double Reference
 - 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 Overriding
- **Demonstrations**
- Remediation
- Summary
- Q&A

Demonstrations – Configuration Item

- Allocating the Object
- De-Allocation the Object

Demonstrations – Configuration Item

- Allocating User Data

Demonstrations – Configuration Item

- Executing a VFunc

Demonstrations – Configuration Item

- Putting it Together
 - De-Allocate
 - Re-Allocate
 - Execute

Demonstrations – Remote Exploit

- Another Exploit on IIS,
but this time – a remote one

Where are We

- What is a Dangling Pointer
- Code Injection
- Object Overriding
- Demonstrations
- **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
- Demonstrations
 - Configuration Item
 - Remote Exploit
- Dangling Pointer
 - Only Object Oriented Objects

Questions

- Ask Away...