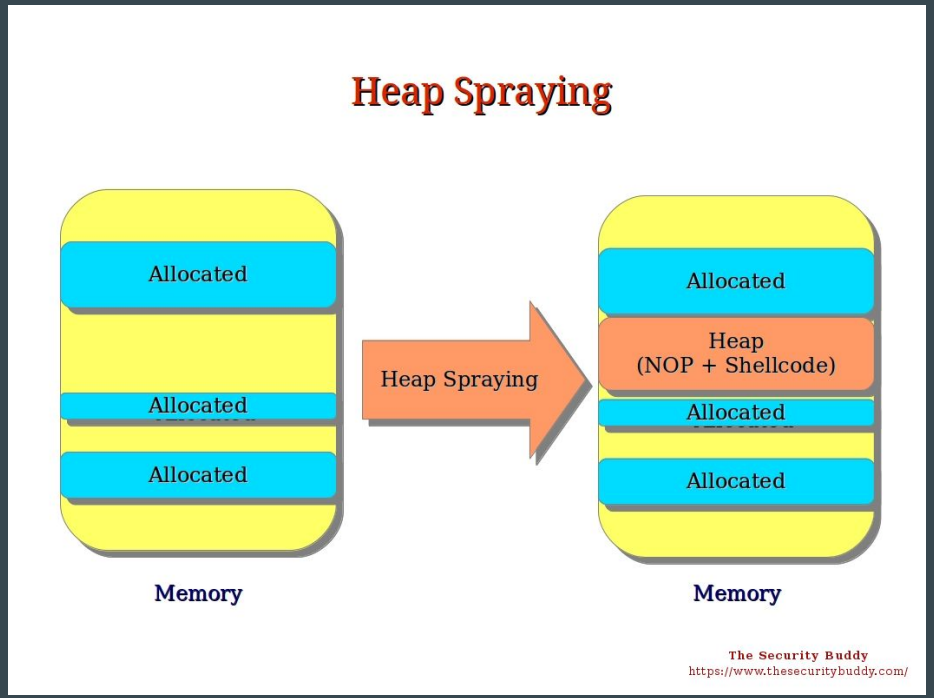
Heap Spraying

Albert Yeung, David Dvorkin, and John Rocco

What is Heap Spraying?

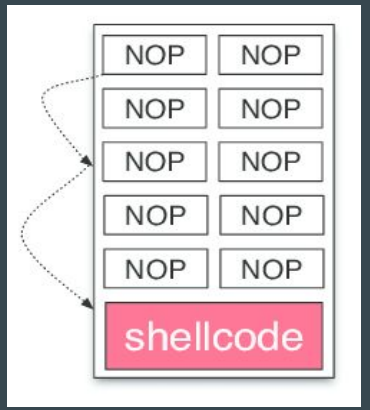
- Populating a program's heap section with malicious code (shellcode)
- A code injection technique; not an exploit on its own
- Spraying a larger area in the heap increases the likelihood of attacker's code being executed
- Gets around address space layout randomization



https://www.thesecuritybuddy.com/vulnerabilities/what-is-heap-spraying/

What is a NOP slide? (NOP sled)

- NO-OP instruction (0x90 in hex) simply goes to the next instruction once read
- A NOP slide is a LOT of NO-OP's stringed together
- If you actually look at the heap, it's easy to see
- If it's easy to see, why use a NOP slide?
  - Without it, your shellcode only has a small entry point, possibly even only one address
  - The larger the NOP slide the larger the area of entry



https://www.researchgate.net/figure/NOP-sled-and-shellcode-appended-to-the-sequence_fig1_225103556

What is shellcode?

- Shellcode is the binary code with malicious intent
- Binary code seen when you see when you objdump a file
- Malicious code must exist in CPU's native instruction set on heap - doesn't need compilation

```
/*
void f3(int* p){
    *p = 12;
}

00000000004012ff <_Z2f3Pi>:
  4012ff:   55                      push   %rbp
  401300:   48 89 e5                mov    %rsp,%rbp
  401303:   48 89 7d f8             mov    %rdi,-0x8(%rbp)
  401307:   48 8b 45 f8             mov    -0x8(%rbp),%rax
  40130b:   c7 00 0c 00 00 00       movl   $0xc,(%rax)
  401311:   90                      nop
  401312:   5d                      pop    %rbp
  401313:   c3                      retq

  */
```

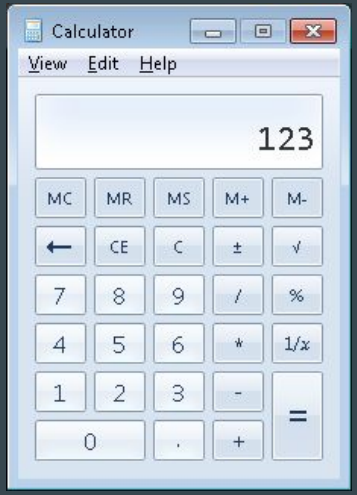
Shellcode execution example

```
#include <cstdio>
#include <stdio.h>
#include <iostream>
using namespace std;

const char shellcode[] = "\x55\x48\x89\xe5\x48\x89\x7d\xf8\x48\x8b\x45\xf8\xc7\x00\x0c\x00\x00\x00\x90\x5d\xc3";

int main(){
    int * testInt = new int();
    std::cout << *testInt << std::endl;
    void (*intChanger)(int*);
    intChanger = (void(*)(int*))shellcode;
    intChanger(testInt);
    std::cout << *testInt << std::endl;
}
```

```
albert@DESKTOP-CQJE7U4:/mnt/c/Users/Albert/Documents/homework/cs458/hea
pspray$ g++ vulnProgram.cpp
albert@DESKTOP-CQJE7U4:/mnt/c/Users/Albert/Documents/homework/cs458/hea
pspray$ ./a.out
0
12
```

Shellcode Demonstration

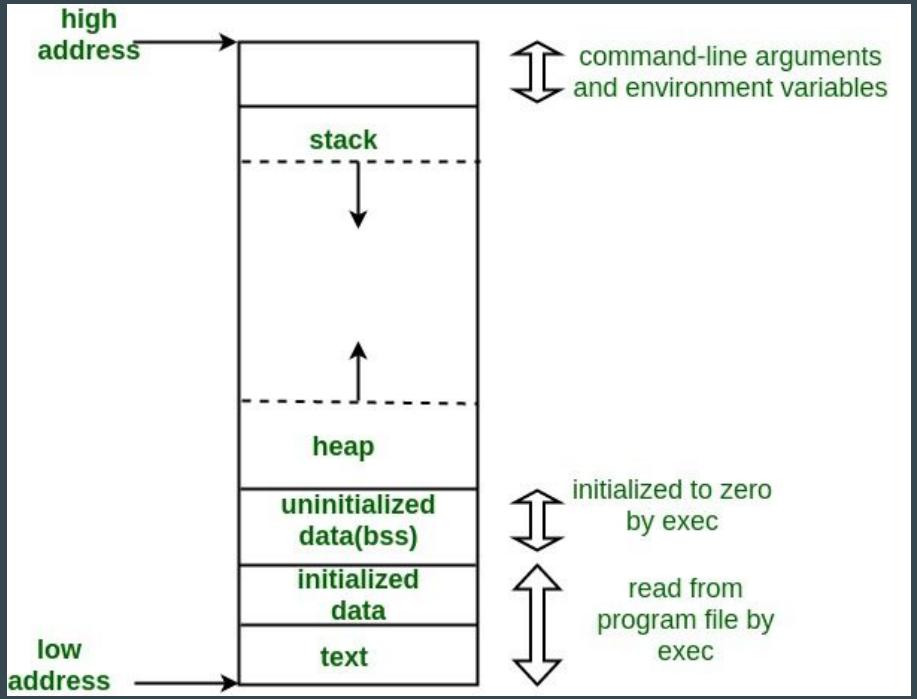
- Named shellcode as it typically starts a command shell where attacker can control anything in the machine
- Can create any malicious c-code executable, objdump, and extract opcodes to create shellcode string to inject
- Demonstrate ability to create new process - calc.exe
- Can reboot, download malware, upload files, etc.



http://chriskonigesminicoding.blogspot.com/2013/12/csharp-c-user32dll-change-focus-to.html

How does heap spraying work?

- Use script to ‘spray’ NOP-sled + shellcode onto the heap section
- Use dynamic memory allocation functions (malloc or keyword ‘new’)
- Then find an exploit in program/browser to trigger a heap read and shellcode will execute
- Heap location usually predetermined
- Takes advantage of writes often being stored consecutively - JS typed array



https://www.geeksforgeeks.org/memory-layout-of-c-program/

Heapspray Pseudocode

```
Create nopsled: large string of no-operations

(if instruction pointer hits anywhere in nopsled, continues down to shellcode)

Create shellcode: x86 instruction string to perform malicious action

spray = new Array();
for(i = 0; i < 100; i++) {
  spray[i] = nopsled + shellcode;
}
```

History of Heap Spraying

- First seen in early 2000s
- Popular in many exploits on IE6
- Drive-by download attacks: malicious web page puts code in browser's memory
- Webpage can't access user's files, but malicious code on the heap that is accidentally run can
- Simple concept, different attacks can reuse code with minor modifications

https://techcrunch.com/2013/10/01/internet-explorer-6-market-share-finally-falls-under-5/

Recent Use

- Attack on Google (2010)
  - China-based; hacked into two accounts, but nothing private stolen
  - Mostly target Chinese human rights activists
  - “Highly sophisticated and targeted attack” - Google blog post

Countermeasures

- The Nozzle Project
  - Microsoft, Established 2008
  - Runtime monitoring for attempts to spray heap
  - Zozzle, Rozzle

- BuBBle
  - Routine that prevents heap spraying
  - Prevents allocations of nop sled + shell code
  - Implemented in Firefox

Thank you!