

Operating Systems Security

04 NOV 2019

Gigamon[®]



Introductions



William Peteroy



@wepIV

Used to hack stuff—now build
things and security research



Security

Our Definition of Security

- Engineer and build systems to prevent attackers from doing things we don't want them to do

Why should anyone care?

The weapon's target was Ukraine. But its blast radius was the entire world. "It was the equivalent of using a nuclear bomb to achieve a small tactical victory," Bossert says.

construction company Saint-manufacturer Reckitt Bencki costs. It even spread back to Rosneft.

The release of NotPetya was an act of cyberwar by almost any definition—one that was likely more explosive than even its creators intended. Within hours of its first appearance, the worm raced beyond Ukraine and out to countless machines around the world, from hospitals in Pennsylvania to a chocolate factory in Tasmania. It – crippled multinational companies including Maersk, pharmaceutical

Business

Google uncovers 2-year iPhone hack that was 'sustained' and 'indiscriminate'

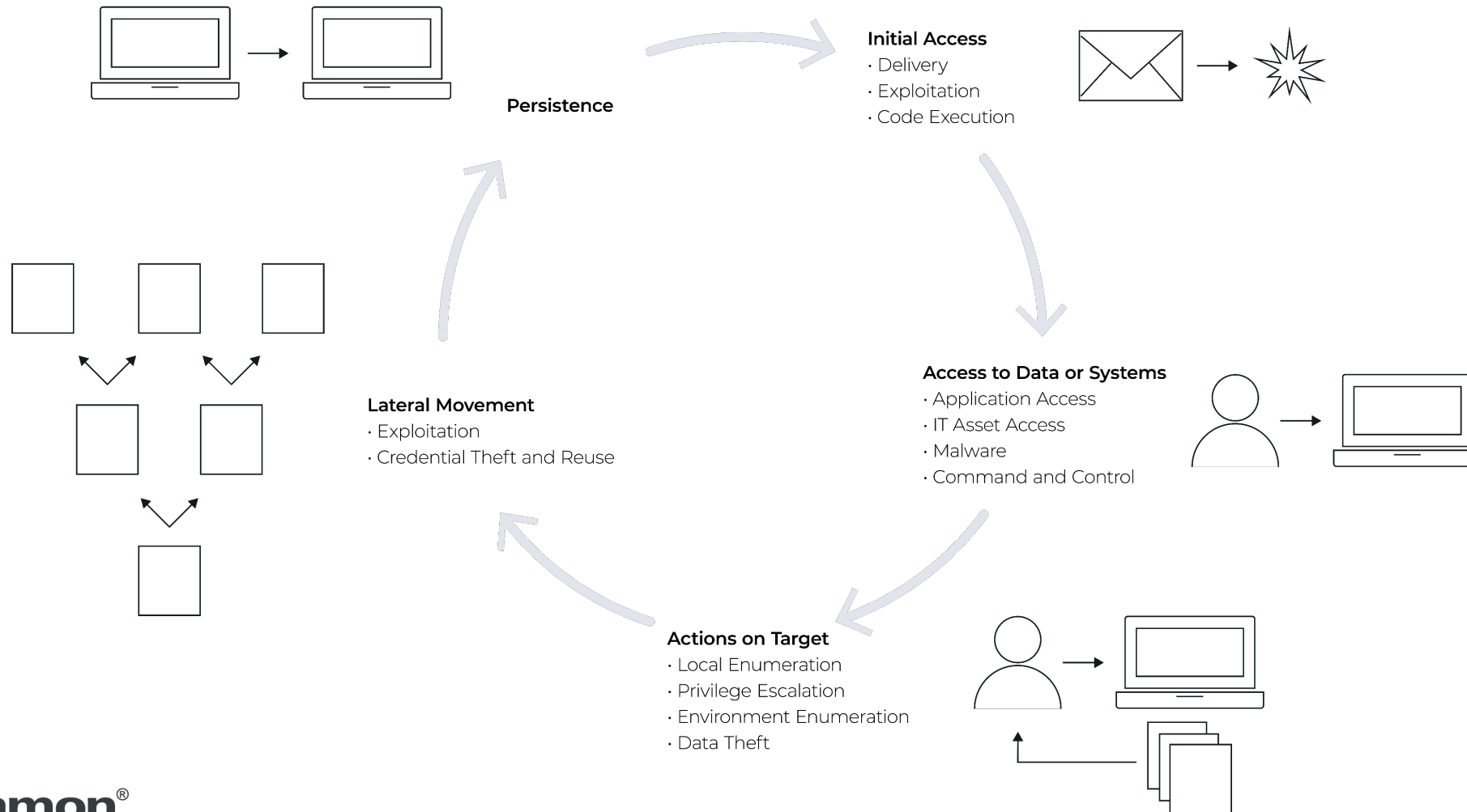
The attack may be one of the largest ever on iPhone users, exposing messages, address books, GPS data and more.

“The result was more than \$10 billion in total damages...”



Attacker Lifecycle

Attacker Lifecycle



What can we do to disrupt the attacker lifecycle?

- Prevent attackers from exploiting systems
- Prevent malicious code from executing
- Ensuring that attackers do not have full (administrative / root) access to compromised assets
- Prevent attackers from moving from one asset to another

What does this mean for OS Security?

- Can we prevent software vulnerabilities?
 - We can try, but ultimately, no
- Can we prevent users from clicking on things
 - We can try, but ultimately, no
- Can we design systems that are resilient to exploitation?
 - Yes, absolutely
- Can we try to keep users and admins separate?
 - Yes, absolutely



Operating Systems Security

Core OS Security Goals

1. Build and enforce security boundaries
 2. Keep secrets secret (passwords, cryptographic keys, etc.)
- “CIA Triad”
 - Confidentiality
 - Integrity
 - Accessibility



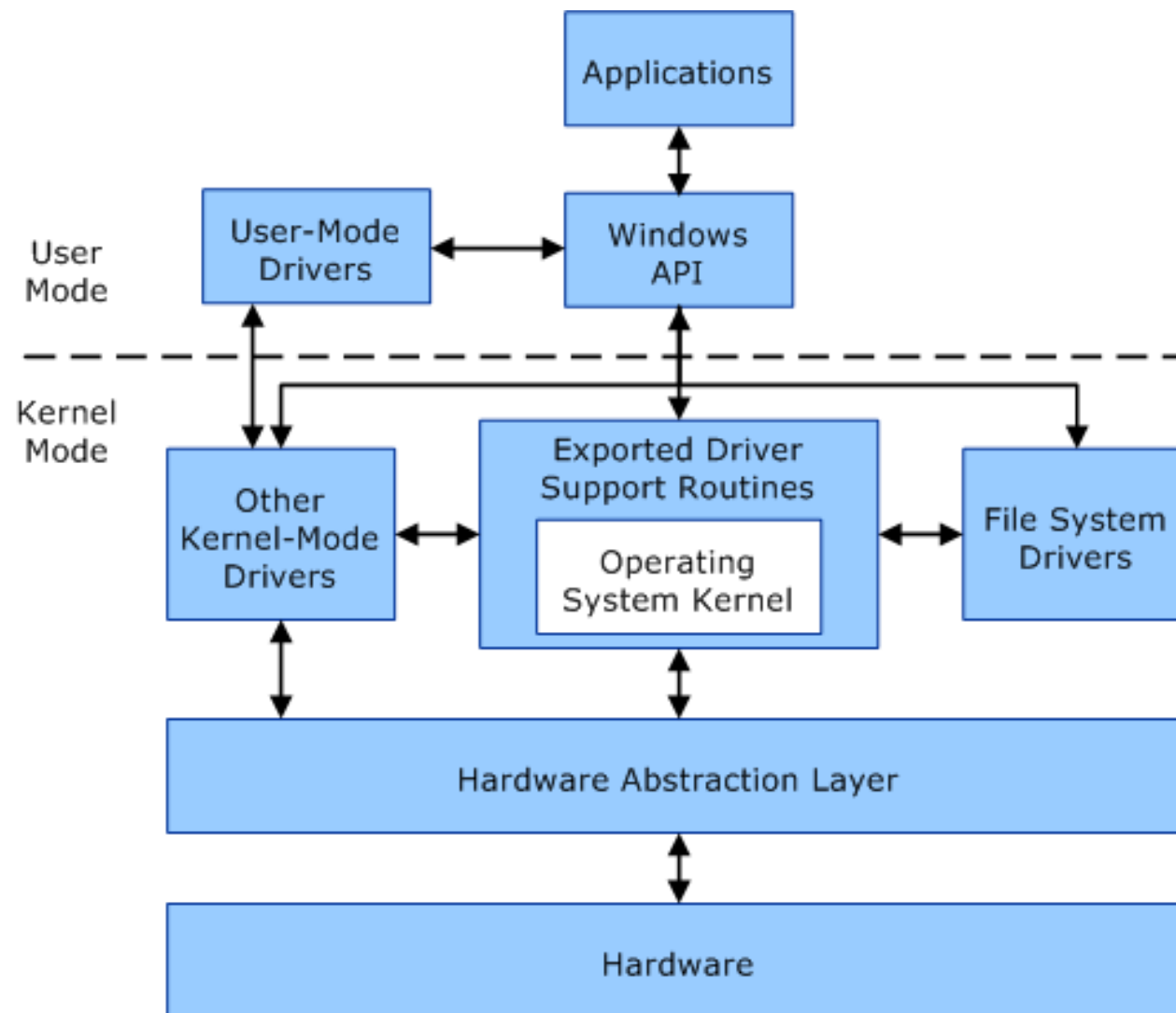
Security Boundaries

What's a Security Boundary?

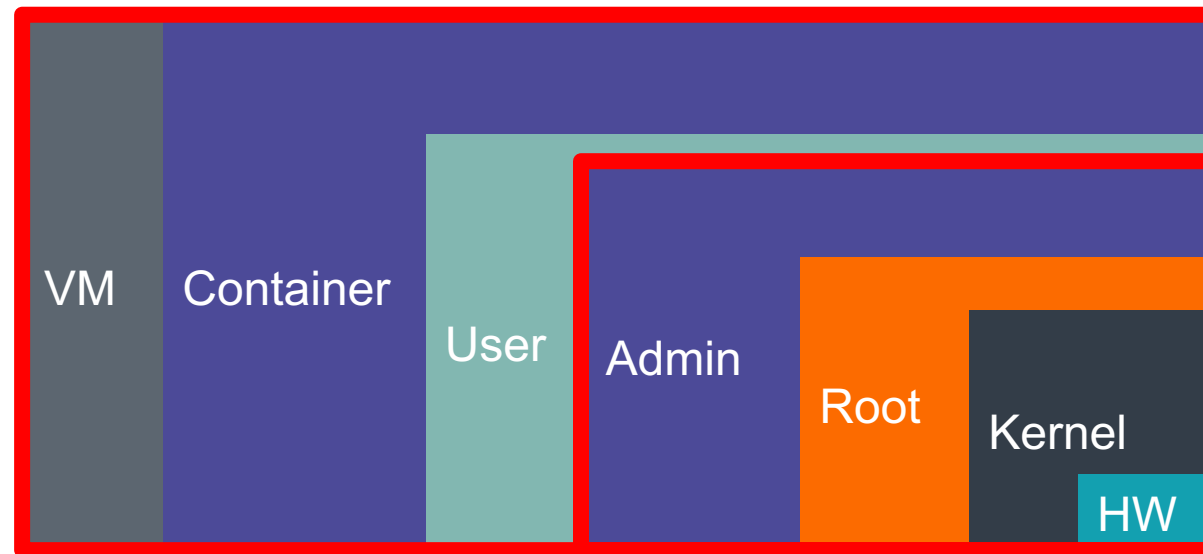
- A Security Boundary is a by-design feature that will be serviced by the OS maintainer / vendor to protect the C/I/A of the Operating System
- Not all things that sound like security boundaries are security boundaries
 - “Secure Boot”
 - “AppLocker”
 - “User Account Control (UAC)”
- <https://www.microsoft.com/en-us/msrc/windows-security-servicing-criteria>

User Mode vs. Kernel Mode

Let's look at Windows

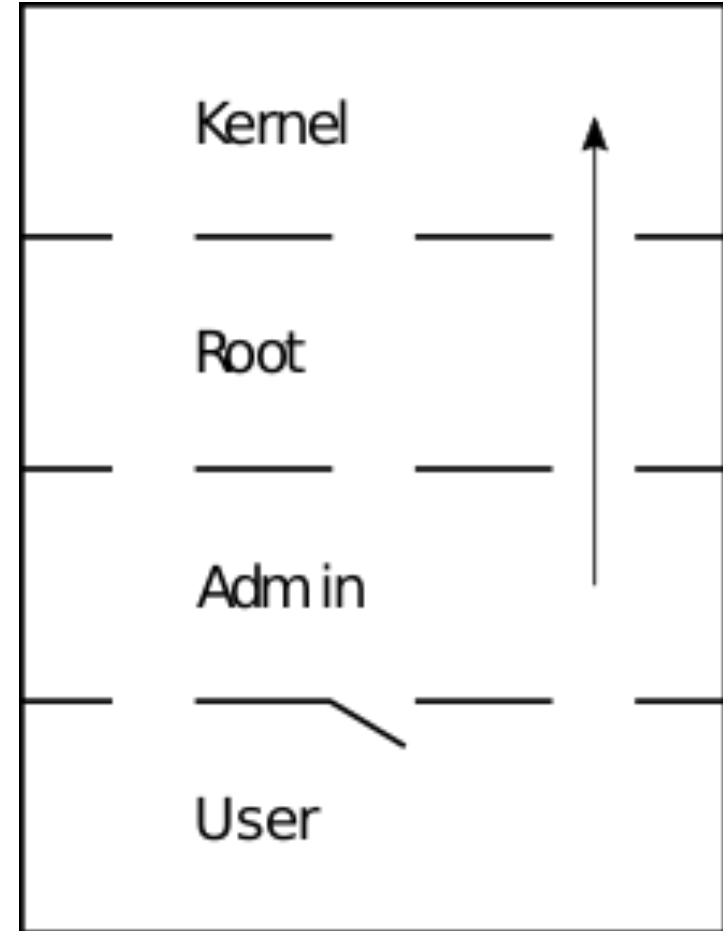


What does this look like in the real world?



Root vs. User

- Who watches the watcher?
- Kernel enforces permissions and access
- Kernel controls direct hardware access
- Boundary between root / kernel / admin
- Elevation of Privilege attacks



Impact of Security Boundaries in Modern Environments

Provide Security:

- Operating System Protections
 - Admin Access
- Virtual Machines
 - Resource Isolation

Do **NOT** Provide Security:

- Containers
 - Do not isolate resources
 - Docker, Kubernetes, etc.
- Admin vs. Kernel
 - Not a security boundary in the real world



Challenges in OS Security

Core Challenges in OS Security

- Performance tradeoffs
- Sandboxing (how do we share system resources / control access to system APIs)
- Secrecy vs. Simplicity
- Wrangling users to make good choices



Performance Tradeoffs

How bad design decisions created the least secure driver on Windows



Thomas Garnier [Follow](#)

Aug 21, 2016 · 5 min read

This driver is called win32k, it manages the user interface of Windows. This post will discuss the multiple bad ideas that are part of this driver.

How bad is it?

It is hard to get a bug count estimate. Each page unique and it can be hard to infer affected modules. Patch and look at the files, when the links still v

Designed with trust in mind

In Windows NT 3.5, the UI was managed by a user-mode module in the CSRSS system process. The original design was overhauled in Windows NT 4 due to bad performance. The win32k driver was created almost as an extension of its user-mode counterpart with an obvious trust between the two of them. This new design was much faster and flexible though hard to make reliable and secure.

Sandboxing

- Maybe we can just have a gatekeeper for all of the resources?
- Can be effective as a mitigation but not as a true prevention measure (Google Chrome)
- Isolation-based approaches like a VM are generally more secure



Secrecy vs. Simplicity

- In general security is good
- Some security can add unnecessary complexity



Dino A. Dai Zovi
@dinodaizovi

Follow



In general, software engineering is all about managing complexity. Security almost always adds complexity. Learning how to add the right amount of security, in the right places, with minimal additional complexity is what most helps the product/business succeed.

7:08 AM - 2 Nov 2019

36 Retweets 101 Likes



2



36



101

User Wrangling

- We can make things secure by default...
- ... up to a point

What is Protected View?

Excel for Office 365, Word for Office 365, PowerPoint for Office 365, Excel 2019, More...

Files from the Internet and from other potentially unsafe locations can contain viruses, worms, or other kinds of malware that can harm your computer. To help protect your computer, files from these potentially unsafe locations are opened as read only or in Protected View. By using Protected View, you can read a file and see its contents and enable editing while reducing the risks.

NEWS

Email attacks exploit unpatched Microsoft Word vulnerability

Attackers have been exploiting a zero-day vulnerability in Microsoft Word since January to infect computers with malware



Quick Primer on Exploitation

Smashing the stack for fun and profit

- Life was great in 1996
- Phrack magazine issue 49

.oO Phrack 49 Oo.

Volume Seven, Issue Forty-Nine

File 14 of 16

BugTraq, r00t, and Underground.Org
bring you

XX
Smashing The Stack For Fun And Profit
XX

by Aleph One
aleph1@underground.org

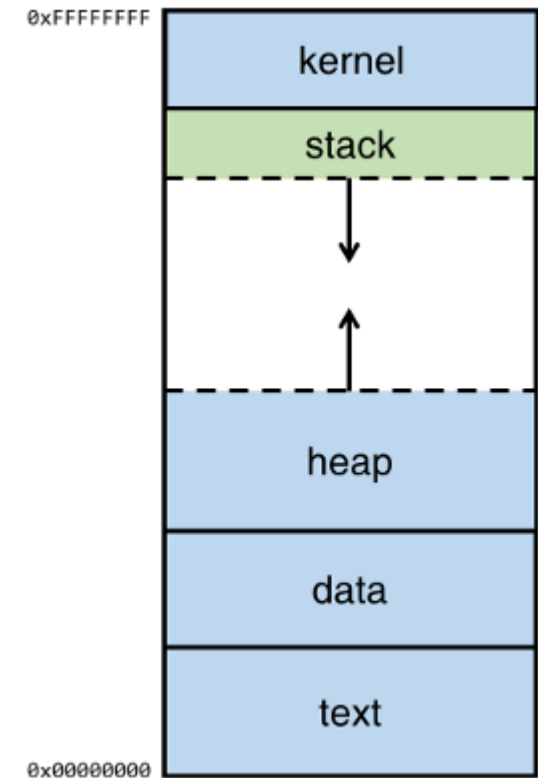
`smash the stack` [C programming] n. On many C implementations it is possible to corrupt the execution stack by writing past the end of an array declared auto in a routine. Code that does this is said to smash the stack, and can cause return from the routine to jump to a random address. This can produce some of the most insidious data-dependent bugs known to mankind. Variants include trash the stack, scribble the stack, mangle the stack; the term mung the stack is not used, as this is never done intentionally. See spam; see also alias bug, fandango on core, memory leak, precedence lossage, overrun screw.

Introduction

~~~~~

# Smashing the stack for fun and profit

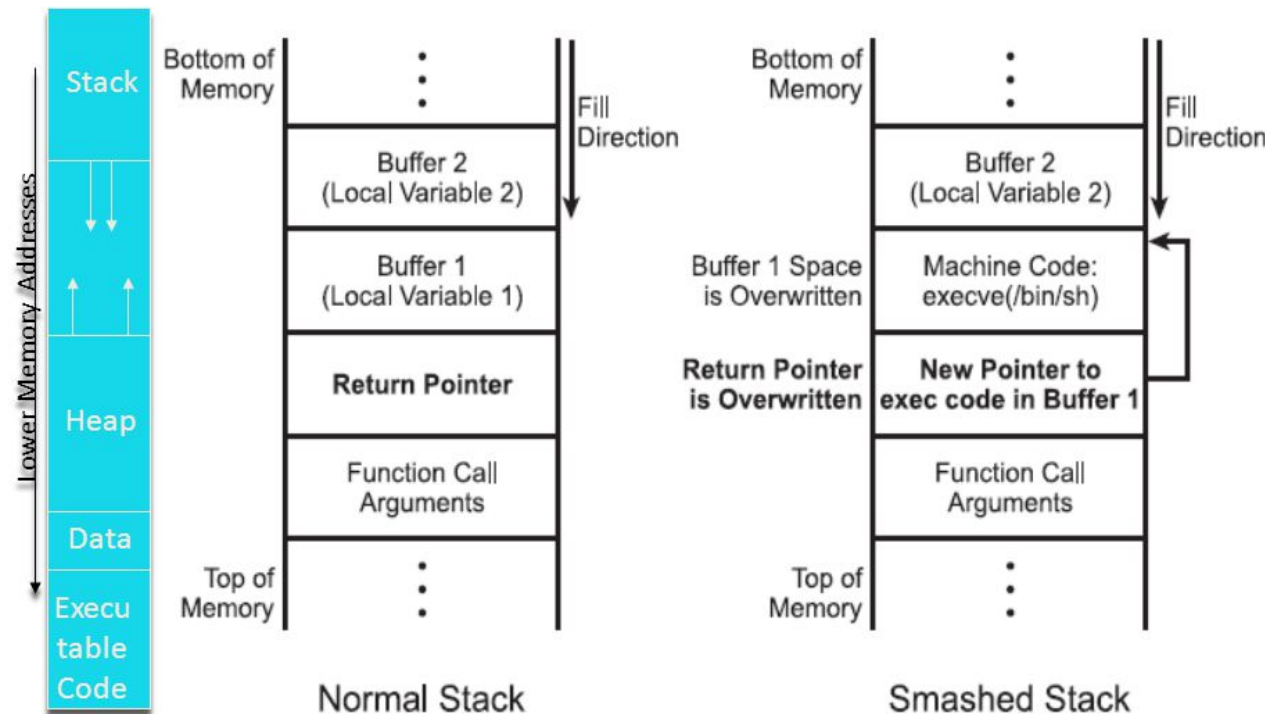
```
void vulnerable() {  
    char buffer[100]; // read string from stdin  
    scanf("%s", buffer);  
    do_something_with(buffer);  
}
```

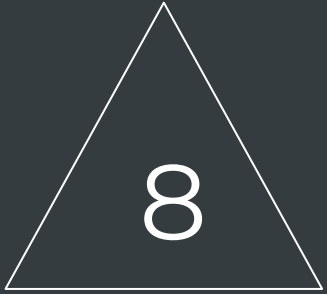


# Memory Corruption

- Things are fun when you overwrite the buffer
- Real fun starts when the function returns

## Stack-Based Buffer Overflows





# Case Study 1

## OS Mitigations

# OS Mitigations

- <https://msrc-blog.microsoft.com/2010/12/08/on-the-effectiveness-of-dep-and-aslr/>
- DEP and ASLR
  - Prevent any code in memory from being executable
  - Prevent attackers from knowing where things are in memory



# LDRHotPatchRoutine

- MS13-063
  - SharedUserData ... exist(ed) at a fixed location in every user process
  - LDRHotPatchRoutine... loads a DLL that's passed as a parameter
- Enabled VUPEN to easily weaponized exploits

```
0:000> dds 7ffe0340 Lc
00000000`7ffe0340 77829ce9 ntdll!LdrInitializeThunk
00000000`7ffe0344 77800100 ntdll!KiUserExceptionDispatcher
00000000`7ffe0348 77800028 ntdll!KiUserApcDispatcher
00000000`7ffe034c 778000b8 ntdll!KiUserCallbackDispatcher
00000000`7ffe0350 7788f8d4 ntdll!LdrHotPatchRoutine
00000000`7ffe0354 77822551 ntdll!ExpInterlockedPopEntrySListFault
00000000`7ffe0358 7782251b ntdll!ExpInterlockedPopEntrySListResume
00000000`7ffe035c 77822553 ntdll!ExpInterlockedPopEntrySListEnd
00000000`7ffe0360 77800190 ntdll!RtlUserThreadStart
00000000`7ffe0364 77892dfd ntdll!RtlpQueryProcessDebugInformationRemote
00000000`7ffe0368 778517d9 ntdll!EtwpNotificationThread
00000000`7ffe036c 777f0000 ntdll!CsrServerApiRoutine
```



# Case Study 2

## Extreme iOS 0days



# iOS Vulnerabilities

- What's the impact?
- What was the root cause?
- Were mitigations successful? Why Not?


# iOS Vulnerabilities

- What's the impact?
  - Nation-state surveillance of an entire population
- What was the root cause?
  - Lots of sandbox issues
  - Graphics parsing
  - Heap allocation
  - Unused code (task\_swap\_mach\_voucher)
  - JIT / JSC bypass
- Were mitigations successful? Why Not?
  - Kind of?
  - We see from this that some exploits didn't need byasses but some did so `^-\"(ツ)\"_/-`



Q and A

# Where to learn / do more with security

- Reverse Engineering
  - CrackMes
- Finding Vulnerabilities
  - Bug Bounties
- Competing / having fun with CTFs
  - “Intro to CTFs”
- My contact info
  - @wepIV 

# Thank You