Michael Weaver

CST – 221

John Zupan

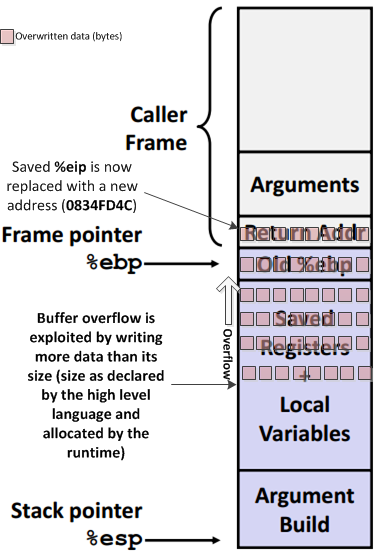
3/4/2018

Github Link: <https://github.com/battousairurik/CST-221>

**Security**

*Buffer Overflow*

Buffer Overflow is when a process attempts to write more data to memory than the buffer allows. This additional data can overwrite data values in memory addresses adjacent to the destination buffer. This tends only to happen when buffers lack bounds checking or the ability to disregard data chunks that are larger than intended.

 Buffer Overflow can overwrite saved data that is nearby the buffer location. For example, user input that is greater than the buffer can overwrite a nearby saved password and cause serious issues with your program. This can occur easily in C because it is a lower-level language. C allows for direct memory access, which in turn allows for easily overwritable data. Additionally C has weak object typing, making it easier for overflows to occur.

Preventing Buffer overflow can be accomplished by using safer functions, ones that only write to the maximum size of the target buffer. When buffer overflow does occur, C compilers contain something called a Canary. These are placed on the stack after each buffer and compared to their original size, catching when buffer overflow occurs. They then flag the system to halt, optimally preventing other harmful outcomes. Additional defenses exist in the form of non-executable stacks and address space layout randomization, which help prevent code injection.

*Zero Day Exploit*

Zero Day Exploits are security vulnerabilities that are exploited before they are discovered. This can take the form of software bugs like buffer overflow, improper networking, or other improper software development. Generally, Zero Day Exploits need to be patched right away to remove these vulnerabilities, though it is up to the user to install the patch. Security Researchers that find these exploits withhold their findings for a set amount of time before publishing so that the developers can patch the vulnerabilities. Given the nature of Zero Day Exploits, they are extremely difficult to prevent. One useful technique would be monitoring user behavior, as highly suspicious activity might very well be an exploit occurring. Zero Day Exploits are unique because developers never know how their programs will interact with the infinite other programs and networks in use.

Zero Day Exploits are highly unethical for all those seeking personal gain from them. In a sense they simply are security vulnerabilities and have no ethical weight whatsoever, but when a person chooses to exploit the vulnerability and steal, that is when unethical nature enters. On the other hand, researchers go out of their way to find these vulnerabilities to report them, which is the right and ethical approach to these Zero Day Exploits. Learning of a Zero Day Exploit can potentially make you rich, stolen information can be black marketed or ransomed to the highest bidder, a truly despicable individual could even bring the exploit to the developer’s attention after hacking the information and charge a fee for assisting in finding the vulnerability. Specific to the Christian worldview, using a Zero Day Exploit goes against the very meaning of work. Given that work is meant to pay homage to God, by exploiting the work of others you are sinning against them, yourself, and God. You are disgracing the very definition of work.

*Kali Linux*

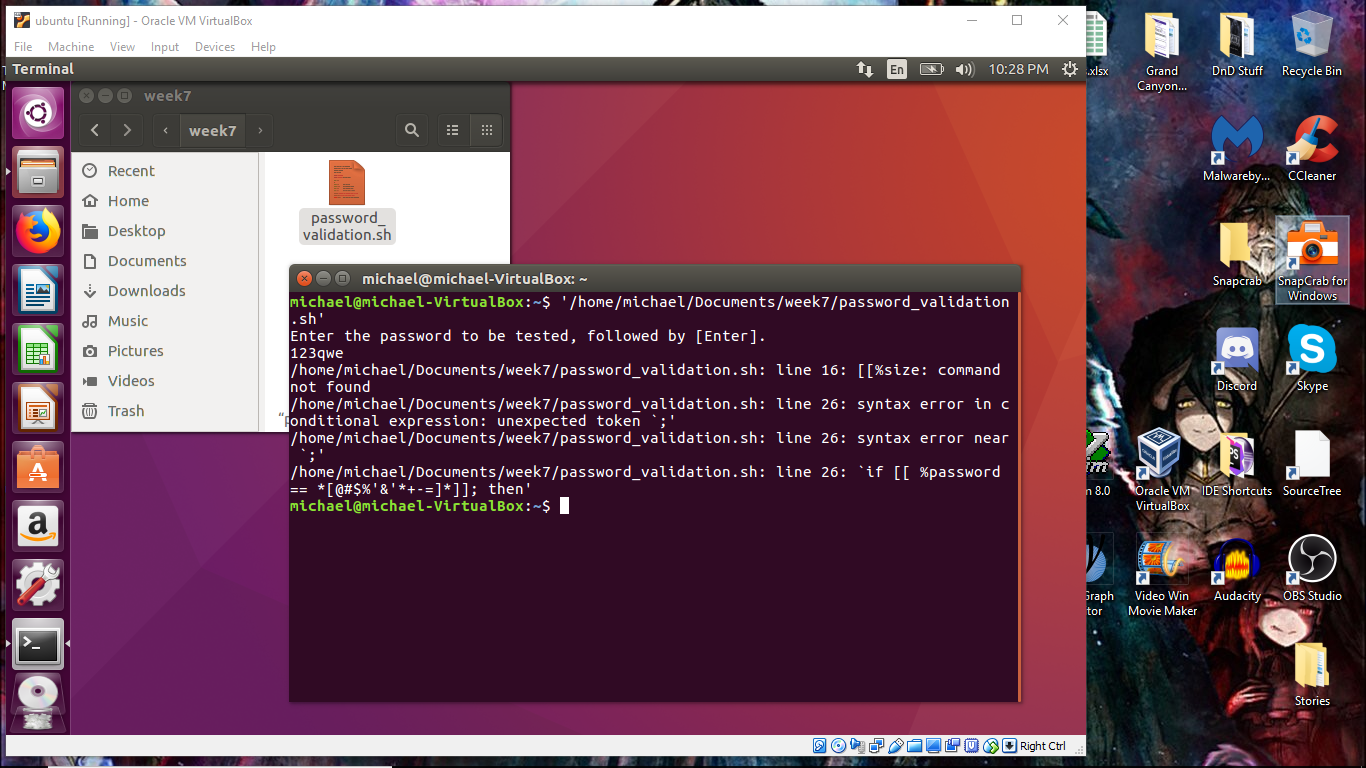
Kali Linux is a distribution aimed at advanced Penetration Testing and Security Auditing. Kali contains hundreds of tools built for Penetration Testing, Security research, Computer Forensics and Reverse Engineering. Developers using Kali can test their systems, projects, etc. in a safe environment. Kali is specifically built for training Linux based security professionals. This is one of the drawbacks of using Kali, its highly system specific, and Kali itself is extremely restrictive with what can be done. For example, you cannot even install Steam (a popular gaming platform) on the Kali system, as it would cause numerous complications. The impression given by Kali Linux is that it is a highly focused platform that cannot be used for anything other than Linux specific security analysis, and only if you already have a technical knowledge of the tools included given the provided documentation is difficult to comprehend.

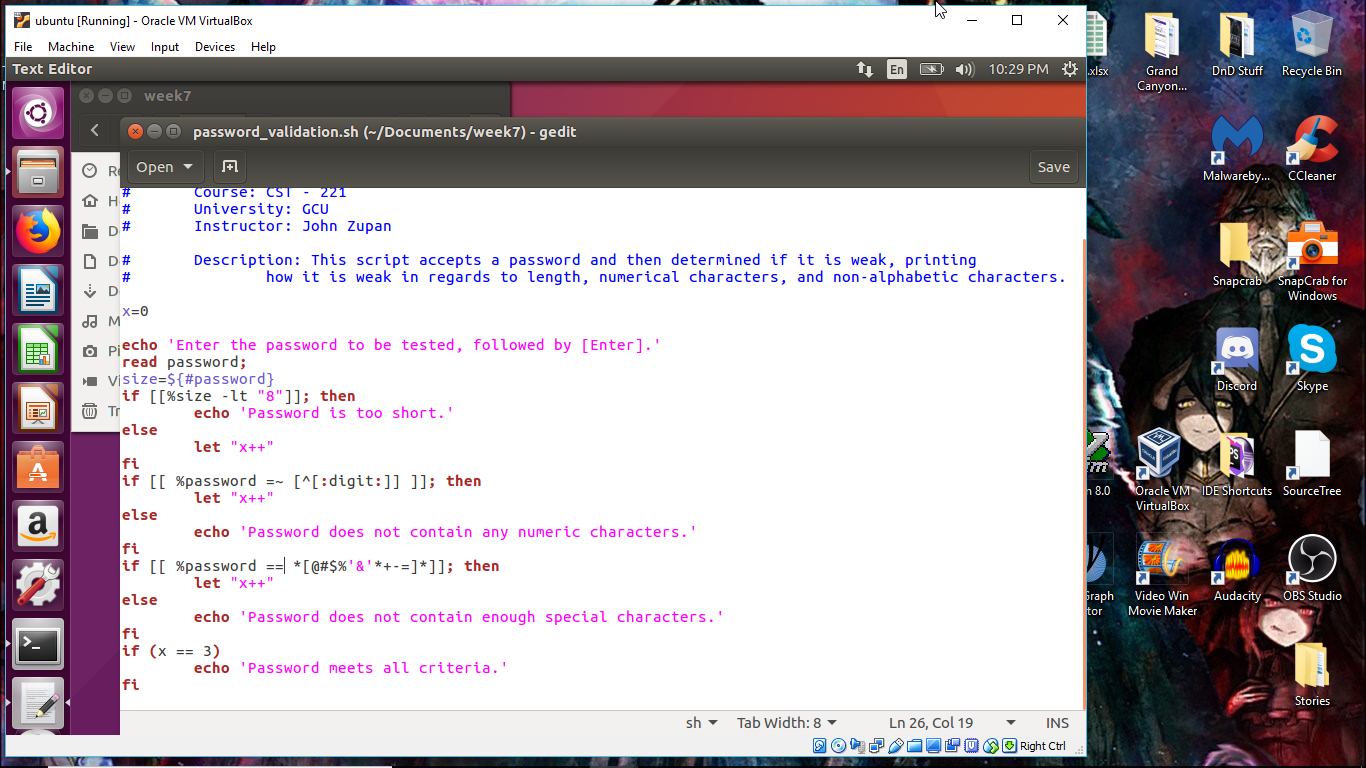
\*Note: There are over 600+ tools in Kali, therefor I will only be discussing the general themes that each tool comes typed as.

|  |  |  |
| --- | --- | --- |
| **Tool** | **Function** | **Use in Cyber Security** |
| Information Gathering | Tools in Kali designed to gather data, such as available hosts, ports, firewalls, etc. | These tools are used to determine what kind of system exploits are possible within the target system. |
| Vulnerability Analysis | Tools designed to target well know vulnerabilities using techniques such as mass scanning and fingerprinting. | These tools are good for routing out easily discoverable vulnerabilities. Once all well-known vulnerabilities are patched then the developer can really focus on the hidden ones. |
| Wireless Attacks | Tools designed to attack wireless connections, intrusion detection systems, create fake access ports, or even attack multiple networks in succession. | These tools can be used to determine if there are leaks in the network or exploits with network interaction. |
| Website Penetration | Tools designed to test the vulnerability of websites and web applications. | Uses techniques like validating SQL injection or Cross-site scripting. Also contains programs to detect database vulnerabilities and take them over. |
| Exploitation Tools | Generic exploitation tools for hacking and bypassing authentication protocols. | Tools can be used to hack systems, or learn how to prevent hacks. They can also bypass certain authentication protocols and therefor learn how to properly reinforce protocols. |
| Stress Testing | Tools for determining how much a system can handle, aka stressing it. | These tools gather data such as connections per second or connection timeout, and is used to determine a programs breaking point. |
| Forensics Tools | Tools designed to track various forms of computer footprints and gather data directly from them. | Tools can scan firewalls and capture packets all without network traffic, probes, queries, or anything noticeable. Can be used to secretly gather information about systems |
| Sniffing & Spoofing | Sniffing is when a tool passively reads network traffic. Spoofing is when a machine acts as another, trying to emulate function and communication. | Hackers can use these tools to build false networks based on flow, and white hackers can use them to build better firewalls. |
| Password Attacks | Tools used to attack passwords to determine the correct input. Techniques include guessing, dictionary, rainbow tables, etc. | Password cracking can gather information about how users choose their passwords, hence the creation of rainbow tables. Hackers can use this data to break into accounts more easily, or developers can improve security like through salt. |
| Maintaining Access | Tools used to maintain access to a hacked machine even after it has been powered off or disconnected from the network. | Hackers can use these tools to steal data over a long period of time. Security Analysts can use these tools to further develop bugs and other forms of observation. |
| Reverse Engineering | Tools used to crack the source code of programs by using the program itself. | Hackers can use this to steal secrets of popular programs, although that is perfectly legal. Security analysts can use these tools to improve program security. |
| Reporting Tools | Tools used for gathering and generating data from all processes performed by kali or the target system. | Analysts can use these tools to provide insight to how hackers work and how to better spot application vulnerabilities. |

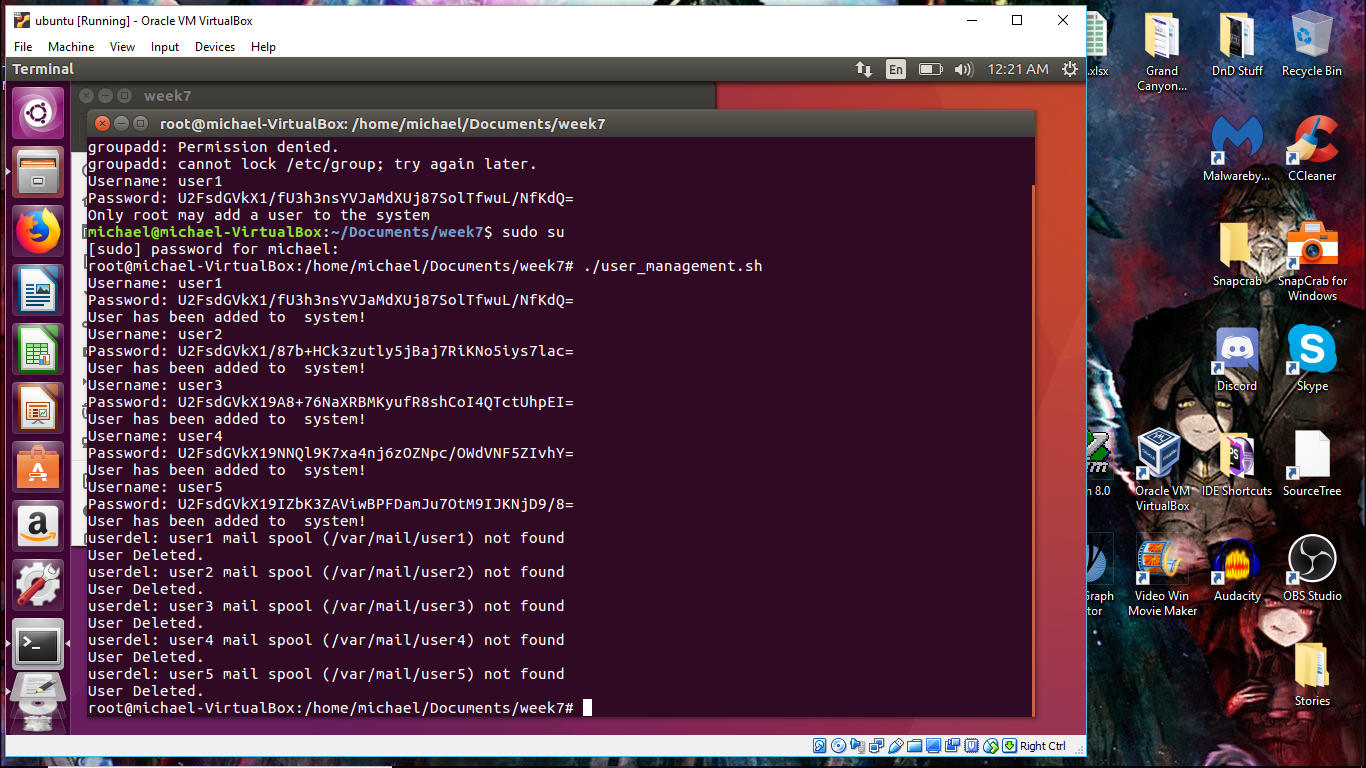
Kali linux can be used for both ethical research and very unethical exploitation. The vast number of tools make it easy to understand system and application vulnerabilities and learn how to exploit them. If you are doing this for research sake, or to understand how to build better programs than it isn’t a bad thing. Though if you are choosing to learn how to exploit systems and programs for personal gain then you are clearly acting in an utmost unethical nature. Worldview speaking, this is simply not something that a Christian should ever entertain because it violates a great number of Gods laws. No matter the perceived good that something might bring, exploiting others, decreasing their overall well-being and quality of life, is sinful and wrong on so many levels. So long as you are using these tools to benefit yourself and others, the right way, then there are no ethical or religious violations.

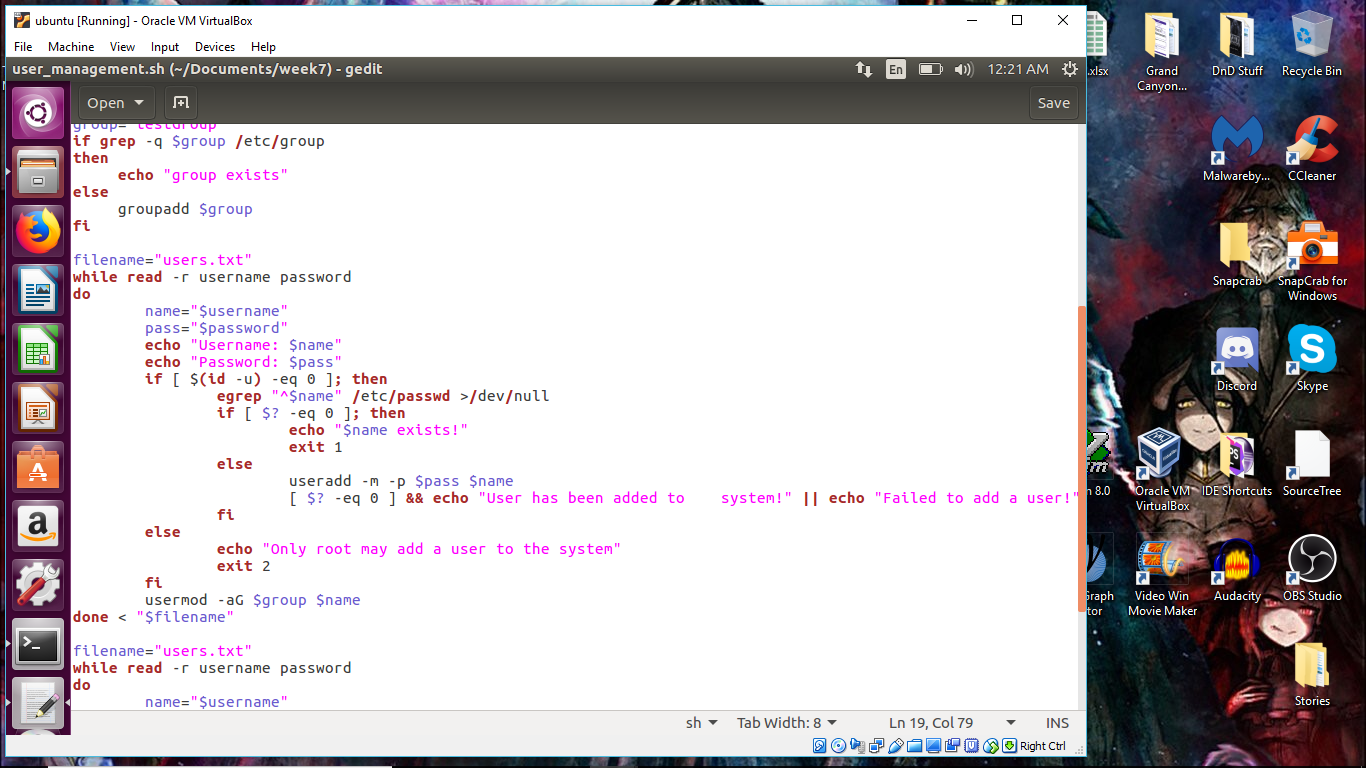
*Screenshot for Testing Password Strength*





*Screenshot User Management Bash Script*





References

Rouse, Margaret. (2016). *Buffer Overflow*. Retrieved from <http://searchsecurity.techtarget.com/definition/buffer-overflow>

Arora, Himanshu. (2013). *Buffer Overflow Attack Explained with a C Program Example*. Retrieved from <https://www.thegeekstuff.com/2013/06/buffer-overflow/>

Kerestan, Benjamin. (2017). *How to detect, prevent, and mitigate buffer overflow attacks.* Retrieved from <https://www.synopsys.com/blogs/software-security/detect-prevent-and-mitigate-buffer-overflow-attacks/>

Rouse, Margaret. (2018). *Zero-Day (Computer).* Retrieved from <http://searchsecurity.techtarget.com/definition/zero-day-vulnerability>

Singh, Shaun. (2017). *What are all the benefits of using Kali Linux*? Retrieved from <https://www.quora.com/What-are-all-the-benefits-of-using-Kali-Linux>

Kali.org. (2018). *What is Kali Linux*? Retrieved from <https://docs.kali.org/introduction/what-is-kali-linux>

Tutorialspoint. (2018). *Kali Linux (all tools pages).* Retrieved from <https://www.tutorialspoint.com/kali_linux/index.htm>

Techiwarehouse.com. (2010). *IP Sniffing and Spoofing*. Retrieved from <http://www.techiwarehouse.com/engine/423a5281/IP-Spoofing-and-Sniffing->

Web.cs.du.edu. (n.d.) *Password Cracking*. Retrieved from <http://web.cs.du.edu/~mitchell/forensics/information/pass_crack.html>

Tldp.org. (2018). *Catching user input*. Retrieved from <http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_08_02.html>

GNU.org. (2018). *Conditional Constructs*. Retrieved from <http://www.gnu.org/software/bash/manual/bashref.html#Conditional-Constructs>

Radeanu, Radu. (2013). *How to increment a variable in bash*? Retrieved from <https://askubuntu.com/questions/385528/how-to-increment-a-variable-in-bash>

TLDP.org. (2018). *Other Comparison Operators*. Retrieved from <http://tldp.org/LDP/abs/html/comparison-ops.html>

Marco. (2012). *Read a file line by line assigning the value to a variable*. Retrieved from <https://stackoverflow.com/questions/10929453/read-a-file-line-by-line-assigning-the-value-to-a-variable>

Mlebel. (2013). *pipe file line by line into multiple read variables*. Retrieved from <https://stackoverflow.com/questions/15442220/pipe-file-line-by-line-into-multiple-read-variables>

Arun Nath. (2012). *Encrypt a string using openssl command line*. Retrieved from <https://stackoverflow.com/questions/10106771/encrypt-a-string-using-openssl-command-line>

nixCraft. (2007). Linux Shell script to add a user with a password to the system. Retrieved from <https://www.cyberciti.biz/tips/howto-write-shell-script-to-add-user.html>