Connor Raymond Stewart
20673233

CS 658 A1

Written Response Questions

1)
   a. Interception, Interruption, Modification, and fabrication attacks:
      i. Interception: An attacker could install a camera logging virus on the interviewer’s phone which captures and transmits the interview footage. The attack assumes the phone is connected to the internet or will be connected to the internet in the future, and that the footage can be saved in a buffer until it can be transmitted.
      ii. Interruption: An attacker could stage an interruption by intercepting or determining the time and location of the interview, then calling authorities with a fake story to stop the interview. The attacker has multiple options to stop the interview, as they could call the police with a fake crime report, or the fire department with a fake fire spotting. The above interruption assumes the attacker has access to information regarding the time and location of the interview, and that they have the means to call authorities to interrupt the interview without being detected. A less physical interruption can occur if an attacker places a virus in the interviewer’s phone which corrupts video recordings, which would stop interviewers from being able to study from interview footage.
      iii. Modification: An attacker could stage a man in the middle attack by intercepting emails sent to participants from the interviewers by changing details regarding the date or time of the interview. We assume that the attackers can intercept emails to and from the participants. Attackers could preform the interception by using viruses planted on the interviewer’s computers or by modifying web traffic to change email messages send from the interviewer.
      iv. Fabrication: An attacker could duplicate the interviewers email address credentials to send participants fake interview information. An attacker could also sign up as a participant and give false information during an interview to mislead the interviewers.
   b. Zoom meeting questions:
      a) Recording the meetings breaks **confidentiality** since it allows unauthorized users to access and gain possession of sensitive information from the meeting.
      b) Zoom bombing breaks **availability** since it prevents people in a zoom meeting from having conversations and presenting information. The Zoom bomber can create enough distractions to seriously disrupt the meetings functionality.
      c) Deleting segments of the interview recordings breaks **integrity** since it alters the information that the interviewer intended to store.

Anyone watching the interview or receiving a copy of it loses access to the information the interviewer intended to show.

2) Five classes of methods to defend against thieves:
    a. Prevent (prevent the attack): An attack during peak hours cannot be absolutely protected against without closing the store or limiting the number of people allowed inside. If there are a lot of people in the store, then it is always possible for a theft to occur no matter how many precautions are taken.
    b. Deter (make the attack harder or more expensive): One way to deter the attack would be to limit the number of people allowed inside the store. If there are less people in the store, it is more difficult for an attacker to go by unnoticed, and a theft would take longer to plan.
    c. Deflect (make yourself less attractive to attacker): Have a poster outside the store mentioning cameras and keep valuable items behind locked glass containers. Attackers would no longer be able to easily steal expensive items anymore and would simultaneously have a higher risk of being caught on tape doing so.
    d. Detect (notice that attack is occurring or has occurred): Have Acrylic Convex Mirrors installed along isles in the store to make it so a cashier at the counter can see what is happening inside the shop from all directions. Alternatively, have the cameras feed live video to the cashier in the store via a TV set at the counter. The effect of having live surveillance in the store is that the cashier can visually detect when thefts are occurring to call the police on attackers.
    e. Recover (mitigate the effects of the attack): A way to recover would be by using theft insurance for the store. If many items go missing during peak hours, the shop owner can claim their insurance to recuperate their losses.
3) Malware classifications, spread, and effects:
    a. sobig: sobig is a computer worm and trojan horse-based piece of malware. sobig infects computers by presenting itself as an attachment in a normal email (trojan), and it uses email attachments and shared network drives to spread between computers (worm). sobig contains its own SMTP agent engine and sends itself to new email addresses gathered from the host computer by searching through files with various extensions for email addresses. sobig effects computers by showing up as an email with one of a number of possible subjects, with a number of possible attachment names.
        - https://www.f-secure.com/v-descs/sobig.shtml
        - https://www.computerworld.com/article/2579931/sobig-worm-getting-bigger.html
    b. WannaCry: WannaCry is a ransomware and worm-based piece of malware, which can spread itself through emails contains zip files send to users as email attachments (meaning it partially spread through a trojan as well). WannaCry spreads by first entering a network via the attached zip files mentioned before, then spreads to other computers in a network by exploiting file sharing arrangements between computers within the network. The effects of WannaCry

are that it uses strong encryption on files within a computer allowing attackers to demand a ransom from users to decrypt the files.
Source(s):

- https://www.dataprotectionreport.com/2017/05/wannacry-ransomware-attack-summary/
- https://www.vox.com/new-money/2017/5/15/15641196/wannacry-ransomware-windows-xp

c. Zeus: Zeus is a trojan malware package which runs on Windows and some mobile devices. It is spread through drive-by downloads and phishing schemes meaning, depending on how it spread, it can also be classified as a trojan since it masquerades itself as regular links for drive-by downloads and email files. The Zeus virus primarily creates a botnet – a group of machines covertly controlled by a command-and-control server under the control of the malware's owner – and steals the banking credentials of the machines it infects. The botnet component allows Zues to steal large amounts of information from users, and it steals credentials for banking websites by keylogging user keystrokes to login to banking websites.
Source(s):

- https://kb.iweb.com/hc/en-us/articles/230268468-Guide-to-Zeus-Infections
- https://usa.kaspersky.com/resource-center/threats/zeus-virus

d. CIH: CIH is a time-bomb based virus (a type of logic bomb) which spreads itself through executable files (making it a trojan) and can replicate itself using executable files shared between computers (which also makes it a worm). The CIH virus activates itself after an infected executable is activated, and it stays in memory to infect other executables accessed by the computer. The CIH infects the free space in memory between files, meaning infected file sizes do not increase (making it hard to detect for antivirus) and it can hook onto file system calls. The virus activates on the anniversary of the Chernobyl (April 26th) and it has two payloads. The first payload overwrites the master boot record, partition table, and file allocation table to overwrite the data stored on the computers drive. The second payload attempts to overwrite the computers BIOS; however, hardware incompatibilities are sometimes present, and the second payload fails.
Source(s):

- https://www.f-secure.com/v-descs/cih.shtml#:~:text=History,spread%20world%2Dwide%20very%20quickly.
- https://malwiki.org/index.php?title=CIH

Programming Questions

sploit1.c (From milestone segment)

The vulnerability found in the stack buffer overflow attack. The buffer stack overflow attack occurs when a program writes data to memory addresses outside of the intended data structure in a program call stack. On line 303 in the pwgen.c file, within the function print_useage, strcat is defined as strcat(buffer, argv[0]). Since argv[0] is the first argument entered into the command prompt at execution, we can overflow the buffer if extra characters are added to the pwgen call from the command prompt. For example, adding encoded string characters o the %x or %s types in front of or behind the call to pwgen can allow this error to occur. The buffer in the program only allocated 512 bytes of information, so we need to write past this limit for the stack-smashing attack to work.

A stack smashing attack can occur by predefining the argv argument in another c file to contain many no operation codes (0x90) before entering the pwgen activation call. When less than two arguments are entered into the program, the strcat code on line 303 is activated so no other arguments can be added; therefore, argv[1] is set to NULL. When excluding the number of bytes to call the function target, 493 bytes of memory are required for an overflow to occur correctly. Another 45 bytes of memory are added to the buffer by adding the shellcode, and another 40 bytes are added by adding the return call, meaning these final areas of memory are not filled with no operation characters. Using gdb on the pwgen executable, we can run the program and use x/200x $esp*/ to determine the address of the return call. The return call address contains a memory address pointing to a memory section with no operation codes.

By creating a large chunk of memory filled with no operation codes, we can essentially create a volume of memory that pushes the memory pointer forward to the shellcode. Essentially, the no-operation keys cause the memory pointer to move forward to the following address, so if the program calls the return call code and moves to the nooperation memory sectors, the memory pointer will 'slide' forward through all the no operation pointers and eventually 'land' at the shellcode in the memory. Essentially, the program calls the shellcode and moves deeper into the buffer, lands on a no operation character, and slides forward through all the proceeding no operation characters until the memory pointer lands on the shellcode.

Alternative coping methods that allow the user to define the number of bytes being copied, like memcpy, could be used since this would allow the user to limit the number of bytes being copied to prevent an overflow. Another method used is strncpy since it lets a user set how many bytes (characters) to copy over. A user could set the size of the buffer minus one to allow the program to only copy as many bytes as the buffer can store. Lastly, if the user is dead set on using strcpy, they can always manually check the size of the string to be copied and abort the program (or manually set a null terminator partway through the string) if the size of the string is to be copied larger then the buffers memory allocation. The last option is inefficient, meaning it would be best to use a different library than strcpy in the case of the pwgen program.

sploit2.c (Non-Functional)

sploit2 is an example of a format string vulnerability located on line 269. fprintf can be used to print the buffer with additional format specifiers like %n being inputted into argv[0]. Since argv[0] is interpreted directly by the fprintf function, we can change memory return addresses to run the shellcode. We can fix the error by using fprintf(stderr, “%s”, buffer) rather then fprintf(stderr, buffer), since this would cause the output in the buffer to be interpreted as a string.

Sploit2 does not work. Currently, the idea behind getting to work would be loading up the env variable with the shellcode information and a NOP-sled, and overriding the return address of the argv[0] to bring the program to the shellcode. Alternatively, the argv[0] could be loaded with the shell code directly, but that approach was not taken. The override address was not found, so the code does not work correctly.

sploit3.c

Sploit3 is an example of a TOCTTOU (time-of-check to time-of-use) vulnerability since the checking of the temp file is completed before the writing of the parameters into the file. A dummy file can be linked onto the temp file used by the pwgen program. The user calling the sploit3 program has permission to access the dummy file, and by linking the temp file to the shadow file and deleting and remaking the temp file, we symbolically link the file to the shadow file. We can then override the shadow file with the contents of the buffer located in the running pwgen program. By adding *root::99999::::::* we can clear the password for the root user such that we can login through a system call to *su root*. We can fix this by using mkstemp() to create temp files. Alternatively, using the programs memory along with RAND_seed(), rather then a temp file could prevent this altogether.

sploit4.c

Sploit4 is an example of an incomplete mediation vulnerability (the program accepts incorrect data from a user) since the get_uid() and get_gid() functions trust that the environment set under the HOME-environment variable contains the home of the current user. If we set the HOME variable to be that of the roots home, the functions will still successfully execute. pwgen can therefore be used to change the root user’s password rather then the current user’s password. By using expect, the sploit can automatically login to the root using *su* by entering the password outputted by pwgen. We can fix this error by using the getuid() and getgid() from unistd.h.