Contents

1 Buffer Overflows

- When a program is executed, contiguous blocks of memory can be allocated to store arrays (buffers)
- If data is written into a buffer that exceeds its size, an overflow occurs
- The data will overwrite the memory beyond the buffer Bad!

1.1 Stack Smashing

- In C and C++, low level functions like strcpy perform no bounds checking at all
- If str is long, we can write into other memory
- By crafting the string str, we can overwrite the buffer and the return address with custom exploit code!
- E.g. unended string, could make it so we read past it, meaning we copy values we shouldn't access to the buffer
- E.g. if buffer is not large enough to fit the string, we might override return address and start execution from another point in code

1.2 Protection: Canaries

- Stack canaries modify the prologue and epilogue of all functions to check a value in front of the return address is unchanged:
- If you can work out the canary value, there is no issue. You could also corrupt the Structured Exception Handler (SEH)

1.3 Data Execution Prevention (NX)

- Modern operating systems (where possible) will mark the stack as non-executable.
- NX on AMD, XD on Intel, XN on arm
- An NX stack means that adding in our exploit code wont work
- We can circumvent this using a return-to-libc attack

1.4 Further Protection

- To defeat ret2libc various 0x0 null bytes are inserted into standard library addresses
- Developers also restrict access to obvious system calls
- Address Space Layout Randomisation (ASLR) moves the address of library and programs around. Every time the program is ran, the stack is moved in address space.
- They dont have to move too much before your hand-crafted ret addresses will break

2 Return-Oriented Programming

Utilizing parts of program in order to generate a sequence of events that would align a stack and push shell scripts on to it.

3 Race conditions

It's a time of check and time of use issue. If we validate something and before we execute it, a program manages to perform a malicious action, there is no way we could catch that.

4 Heartbleed

- Heartbleed is a bug in OpenSSL
- Problem with a buffer overread
- When a message was sent with different request.size and payload.size, server memory would be copied.

Reference section

placeholder